AD-A068 957 NAVAL UNDERSEA WARFARE CENTER SAN DIEGO CA F/G 17/1
CLIPPED REFERENCE CROSS-CORRELATION COMPUTER PROGRAM,(U)
JAN 68 W J WILSTERMAN
UNCLASSIFIED NUWC-TN-44 NL

| OF |
AD
A068957

END
DATE
FILMED
7-79
DDC

AD A068957
001836
DDC FILE COPY

~~#169~~ MOST Project - 3

(14) NUWC-TN-44

LEVEL II

(1)

NW

NAVAL UNDERSEA WARFARE CENTER

(11) JAN 68

(12) 62 p.

Good.

(6) CLIPPED REFERENCE CROSS-CORRELATION COMPUTER PROGRAM,

(10) by W. J. Wilsterman, Jr.
San Diego, California

(16) F10103

(17) SUBPROJECT NO. SF1010316
TASK NO. 11197

DDC
RECEIVED
MAY 24 1979
D

ACCESSION for
NTIS White Section
DDC Buff Section
UNANNOUNCED
JUSTIFICATION
Per Htr. on File
BY
DISTRIBUTION/AVAILABILITY CODES
Dist. AVAIL. and/or SPECIAL
A

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

403 023

Gp 5

001836

TABLE OF CONTENTS

		PAGE
I.	INTRODUCTION	1
	A. Basic Theory of Cross-Correlation	1
	B. Clipped Reference Cross-Correlation	4
	C. Programming Options	4
II.	PROGRAM USAGE	9
	A. Card Deck Preparation	9
	B. Parameter Cards Preparation	10
	C. END Cards	12
III.	PROGRAM OUTLINE: GENERAL DESCRIPTION	12
	A. CARDINPUT	12
	B. TRANTIME	12
	C. READPACK	12
	D. RDUNPACK	12
	E. COMPCORR	13
	F. PRINTOUT	13
IV.	CARDINPUT	13
	A. Translation of First Card Parameters	13
	B. Translation of Second Card Parameters	14
	C. Recognition of End Card; Exit Condition	15
V.	TRANTIME	15
	A. Specific Explanation	15
	B. Compute REFNUM and ECHNUM	15

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

TABLE OF CONTENTS (Continued)

		PAGE
VI.	READPACK	15
	A. Specific Information	15
	B. POSTIME	16
	C. Mag Tape Input Data Format	16
	D. Core Data Format	18
	E. SHOVEMIN	19
VII.	RDUNPACK	19
	A. PASSFORD	20
	B. AREA Table Design	20
	C. Output Function Words	21
VIII.	COMPCORR	22
	A. Continuous Mode	22
	B. Segmented Mode	23
	C. GTFLAG Modes	23
	D. PKFLAG Modes	24
	E. UPDATE S/R	24
	F. GENCORR	25
IX.	PRINTOUT	30
	A. Correlation Value Print-Out	30
	B. Cross-Correlation Graph Print-Out	30
	C. UNPACKTIME	31
	D. SETFMUP	31
	E. GRAPH	31
	F. PRTDATE	32
	G. SETTIME	32

TABLE OF CONTENTS (Continued)

		PAGE
X.	TIMING	32
	A. Data Input	32
	B. Cross-Correlation Time Estimation	32
	C. CC Value Print-Out Time	33
	D. Cross-Correlation Graph Print-Out Time	33
REFERENCES		35
APPENDIX A		36
APPENDIX B		38
	Clipped Reference Cross-Correlation	38
	Clipped Reference Cross-Correlation	39
	Clipped Reference Cross-Correlation	40
	Clipped Reference Cross-Correlation	41
APPENDIX C		42
	CLIPCORR S/R	42
	READPACK S/R	43
	RDUNPACK	44
	COMPCORR	45
	PRINTOUT	50

PREFACE

This memorandum describes a computer program which was developed by the author to analyze sonar data using cross-correlation techniques. This program was written for Code D603, Naval Undersea Warfare Center, San Diego Division, Subproject SF 101 03 16, Task 11197. This memorandum has been prepared because it is believed that the information may be useful in this form to others at the Naval Undersea Warfare Center (NUWC) and to a few persons outside NUWC. This memorandum should not be construed as a report as its only function is to present for the information of others an explanation of the above mentioned computer program.

ABSTRACT

The CLIPREF System is a program which performs clipped reference cross-correlation between two lengths of formatted digitized sonar acoustic data. The program was written by the author for Code D603 in conjunction with the analysis of sea test data. The CLIPREF System has a number of programming options including selection of data, segmented or continuous cross-correlation and optional outputs. It also makes use of certain time-saving computation techniques.

I. INTRODUCTION

A. Basic Theory of Cross-Correlation

Definition of cross-correlation function:

Let $f = f(x)$ and $g = g(x)$ be two integral real functions over the domain $a \leq x \leq b$. Let τ be a small variation of x. The normalized cross-correlation function $\sigma = \sigma(\tau)$ is defined as:

$$\sigma(\tau) = \frac{\int_a^{b-\tau} f(x)g(x+\tau)dx}{\int_a^{b-\tau} |f(x)g(x+\tau)dx} \qquad (1)$$

From the definition of the cross-correlation function, it is clear that the range of $\sigma(\tau)$ lies between -1 and +1. The function is useful in statistics because it is a measure as to how close two functions resemble each other over a given interval. A high correlation value of .8 or greater indicates that f and g are very similar over the interval $a \leq x \leq b$. A value of +1 indicates full positive correlation; both functions always have the same sign. A high negative correlation value of -.8 or less indicates that f and g are similar but opposite in sign. A correlation value of -1 indicates that f and g are exactly opposite in sign throughout the interval $a \leq x \leq b$. Low correlation values from -0.2 to +0.2 indicate that f and g bear little relation to each other. Intermediate correlation values signify a continuum of evaluations from positive correlation to no correlation to negative correlation.

If f and g are distinct functions over $a \leq x \leq b$, then $\sigma(\tau)$ is called the cross-correlation function of f and g over the interval $a \leq x \leq b$. If f and g are identical functions, then $\sigma(\tau)$ becomes

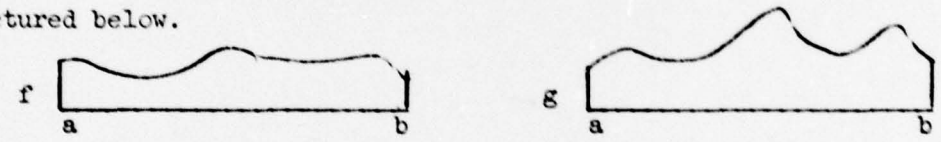
the auto-correlation function of f over the interval $a \leq x \leq b$.

Normalized auto-correlation is defined as:

$$\sigma(\tau) = \frac{\int_a^{b-\tau} f(x)f(x+\tau)dx}{\int_a^{b-\tau} |f(x)f(x+\tau)|dx}$$

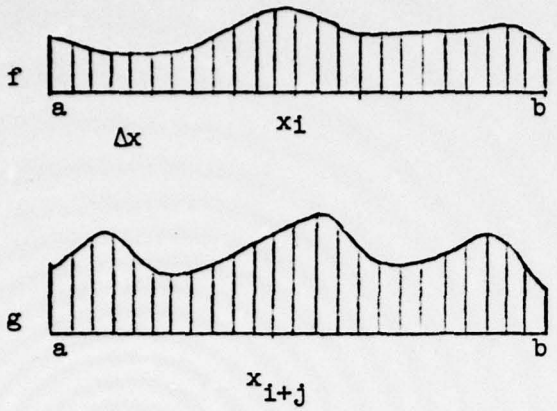
A digital computer cannot handle f and g as continuous arbitrary real-valued functions. It is not possible to define such a function in advance, i.e., a priori, in order to program the computer to perform the integration and to compute the function $\sigma(\tau)$. So in practice, the function $\sigma(\tau)$ is estimated. This is done by estimating first the functions f and g.

Consider the arbitrary continuous real-valued functions f and g as pictured below.



f and g over the interval $a \leq x \leq b$ can be estimated by dividing the interval $a \leq x \leq b$ into a sufficiently small number of equal length segments. The criteria for sufficiently small is determined by the precision of the information desired. For further information on bandwidth and sampling criteria, see "Introduction to Sonar Technology", NAVSHIPS 0967-029-3010.

Now let us divide f and g over $a \leq x \leq b$ into n equal segments. Each segment of equal length will have a width of Δx, $\Delta x = \frac{b-a}{n}$



Consider the following sum:

$$\sum_{i=0}^{n-j} f(x_i)g(x_{i+j})\Delta x \tag{2}$$

Let $x_o = a$ and $x_n = b$ and x_i be the i-th sample of f and x_{i+j} be the i+j-th sample of g. Let Δx be a value such that $\tau = j\Delta x$. Then if we take the limit as $\Delta x \to 0$ of expression (2), we obtain:

$$\lim_{\Delta x \to 0} \sum_{i=0}^{n-j} f(x_i)g(x_{i+j})\Delta x = \int_a^{b-\tau} f(x)g(x+\tau)dx \tag{3}$$

which is the numerator of equation (1). Thus, expression (2) is an estimate of the numerator of equation (1). We can write a similar equation for the denominator. Thus we have, from (1) and (3),

$$\sigma(\tau) = \frac{\int_a^{b-\tau} f(x)g(x+\tau)dx}{\int_a^{b-\tau} |f(x)g(x+\tau)|dx}$$

$$\cong \frac{\sum_{i=0}^{n-j} f(x_i)g(x_{i+j})\Delta x}{\sum_{i=0}^{n-j} |f(x_i)g(x_{i+j})|\Delta x}$$

$$= \frac{\Delta x \sum_{i=0}^{n-j} f(x_i)g(x_{i+j})}{\Delta x \sum_{i=0}^{n-j} |f(x_i)g(x_{i+j})|}$$

$$\sigma(j) = \frac{\sum_{i=0}^{n-j} f(x_i)g(x_{i+j})}{\sum_{i=0}^{n-j} |f(x_i)g(x_{i+j})|} \qquad (4)$$

This last expression is the equation used in the running program to compute the cross-correlation function, $\sigma(j)$.

B. Clipped Reference Cross-Correlation

The CLIPREF system routine is designed to operate in conjunction with formatted digitized data from the FORMATY2 computer program. FORMATY2 is a computer program which processed and formatted the input data used in CLIPREF. The format of this data is described in Section VI, C of this memorandum. The origin of this data is sonar acoustic data which was recorded at sea on analog tapes. A signal of pseudorandom noise or frequency modulated linear sweep was sent into the water. As it was being sent into the water, it was recorded by feedback circuitry onto analog recording tapes. This type of information is called a reference signal or simply a reference. After the signal had been transmitted into the watery medium, the external receiving gates were opened, and sonar information was received by the system from the ocean itself. This information was also recorded on analog tapes for further analysis and is called an echo signal or more simply, a signal. Later, at the laboratory, these echo and reference blocks of data containing analog acoustic data were digitized by the FORMATY2 program and stored on digital magnetic tape, so that this information could be used as data input to analysis programs such as CLIPREF.

Each element of echo or reference data on mag tape consists of 12 bits of signed information. Inspection of Equation (4) indicates that the computation of a cross-correlation value involves a great deal

of multiplication. To multiply two signed twelve bit numbers requires from 35.2 to 112 microseconds on the Univac 1206 computer. Thus to compute just one element of this sum would require at least 110 - 140 microseconds. Straight computation of the cross-correlation function by Equation (4) is excessively time consuming where timing is important.

However, this cross-correlation function can be very closely approximated by clipping either the reference, the echo or both. The effect of the clipping would be a greatly simplified computation and therefore, usually, a greatly reduced computer computation time. Clipping is defined to be that process by which a signal or element of information is so reduced in precision as to be limited to having only two possible values, relative to given thresholds.

If we arbitrarily designate these two values as zero and one and the reference value as p, then we shall assign a data element a value of 1 if greater than or equal to p and a value of 0 if less than p. In this particular case, i.e., in the program CLIPREF, p=0 because the information desired is the sign of reference elements. However, the clipped value assigned is 0 if the element is positive and 1 if the value is negative.

In the CLIPREF program, we are clipping the reference. Thus, if f is the echo and g is the reference, Equation (4) reduces to:

$$\sigma(j) = \frac{\sum_{i=0}^{n-j} f(x_i)g(x_{i+j})}{\sum_{i=0}^{n-j} |f(x_i)|} \qquad (5)$$

where $g(x_{i+j}) = \pm 1$. With Equation (5), the computation of a cross-correlation value can be done in the computer by a comparison of signs

sequence in the numerator and summation and a straight summation in the denominator rather than multiplication in both numerator and denominator. Equation (5) is faster than Equation (4); however, the technique actually used by CLIPREF is much faster than even Equation (5). This is discussed in the section explaining the GENCORR subroutine.

C. Programming Options

There are basically two programming options available: continuous mode and segmented mode.

1. Continuous Mode: Continuous mode cross-correlation consists of the cross-correlation of a short signal with a long signal by sliding the shorter signal along the longer signal from one end of it to the other. There are two ways in which this can happen in CLIPREF: (1) The echo is longer than the reference, (2) the reference is longer than the echo.

We wish to define the following terms: REFNUM is number of reference input samples; ECHNUM is the number of echo input samples; JJ is the number of output cross-correlation values. Accordingly, if REFNUM > ECHNUM, then JJ = REFNUM - ECHNUM + 1, and if ECHNUM > REFNUM, then JJ = ECHNUM - REFNUM + 1. Let E_i be the set of echo signal samples where E_i is the i-th sample of the echo signal and let R_i be the set of reference samples where R_i is the i-th reference sample. The formulas for the cross-correlation functions are as follows:

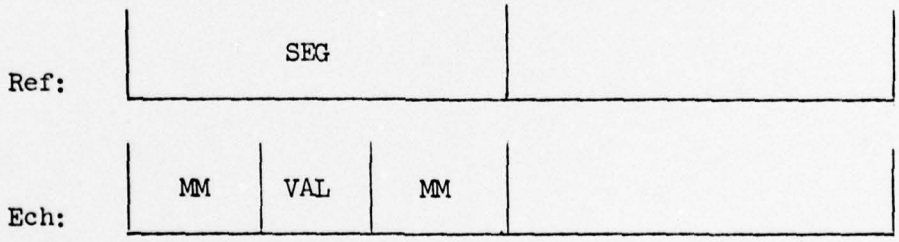
Case #1: REFNUM > ECHNUM

$$\sigma(j) = \frac{\sum_{i=0}^{ECHNUM-1} R_{i+j}E_i}{\sum_{i=0}^{ECHNUM-1} |E_i|}, \quad j = 0, 1, 2, \ldots, JJ - 1 \qquad (6)$$

Case #2: ECHNUM>REFNUM

$$\frac{\sum_{i=0}^{REFNUM-1} R_i E_{i+j}}{\sum_{i=0}^{REFNUM-1} |E_{i+j}|}, \quad j = 0, 1, 2, \ldots, JJ-1$$

2. Segmented Mode: Segmented cross-correlation is the cross-correlation of two signals of equal length by equi-length segments. Each segment is slid a maximum of MM points in each direction and is correlated against its counterpart. QQ is the number of segments into which the input data shall be divided. MM is the maximum number of samples that the echo signal is slid against the stationary reference. The number of output samples per segment, JJ, is equal to 2MM+1. The total number of output samples would be QQ(2MM+1). In segmented cross-correlation, the ECHO segment less MM samples at each end is moved across the reference from the left end at j = -MM to the right end at j = +MM. Consider the following diagram in which QQ = 2.



(8)

In the CLIPREF routine, the reference and echo files must be equal and an integral multiple of QQ. In practice, if they are unequal, the longer file of data is truncated at the end so as to equal the shorter file. Then both are truncated at the ends, if necessary, to make them an integral multiple of QQ. SEG is the number of elements per segment and is computed by SEG = REFNUM/QQ. VAL+1 is

is the number of elements in the echo segment which are correlated with the reference. VAL = SEG -(2MM+1). Each segment is computed by the following equation:

$$\sigma(j) = \frac{\sum_{i=MM}^{VAL+MM} R_{i+j}E_i}{\sum_{i=MM}^{VAL+MM} |E_i|} \qquad (9)$$

where i is indexed from the beginning of the segment. Inspection of (9) reveals that the denominator of the right hand side is a constant with respect to indexing j, and hence need only be computed once.

3. Output Options: There are two parts to the output. The first part consists simply of a listing in order of computation of all the values computed. The values are simply outputted and listed ten to a line in a table similar to a log table. The second part of the output consists of a graph of the cross-correlation function. The graph is precise to 0.02 or to 2%. However, values are printed one to a line; hence the second output is ten times longer than the first. To save time, the second output can be ignored if so desired by setting key 1. If the second output is desired, then clear key 1.

4. Core Option for Reference Input: Input data is read in to the computer memory in two sets. The reference data is read first, then the echo data. Each successive run requires a different set of echo data. However, successive runs may require the same set of reference data. To save time on data input, there is a core option for the reference. If core option is not used, a new run

will read in a new reference file. If core option is selected, the old reference stays in core and only a new echo is read into the computer. Selection of core option is discussed below under parameter cards preparation.

II. PROGRAM USAGE

A. Card Deck Preparation

1. Job Card: Every job at the waterfront computer laboratory, NUWC, begins with a job card. The job card should include the following information: (1) the code making use of the program, (2) the problem number funding the use of the program, (3) the estimated time of run, (4) name of the programmer requesting the run, and (5) the date on which the run was submitted. Job card format is discussed in the WFA COMPUTER CENTER USERS MANUAL.

2. LIB Cards: Two library routines are used in conjunction with CLIPREF, PRINT and LILFLOP. PRINT is a general purpose output routine for such devices as high speed printer, flexowriter and paper tape punch. LILFLOP is a general purpose routine for performing floating point operations. It is used by CLIPREF to format the print-out of decimal numbers for the PRINT package. PRINT is based at 140 and runs almost to 2050. LILFLOP is based at 2050 and runs almost up to 7000. Library cards format is explained in the WFA User's Manual. Load PRINT at 140 and LILFLOP at 2050.

3. LOD40 and TRA Cards: CLIPREF is based at 7000 and runs up to about 20500. The remainder of core to 67777 is used as a general data storage area. Currently CLIPREF is on File 7 of tape WFA-268. When using this system, be sure to mount the program tape on unit M2. After loading CLIPREF and the two library routines, initiate use of the program by transferring to 7000 with a transfer

card. Format of LOD40 and TRA cards is also described in WFA User's Manual.

B. Parameter Cards Preparation

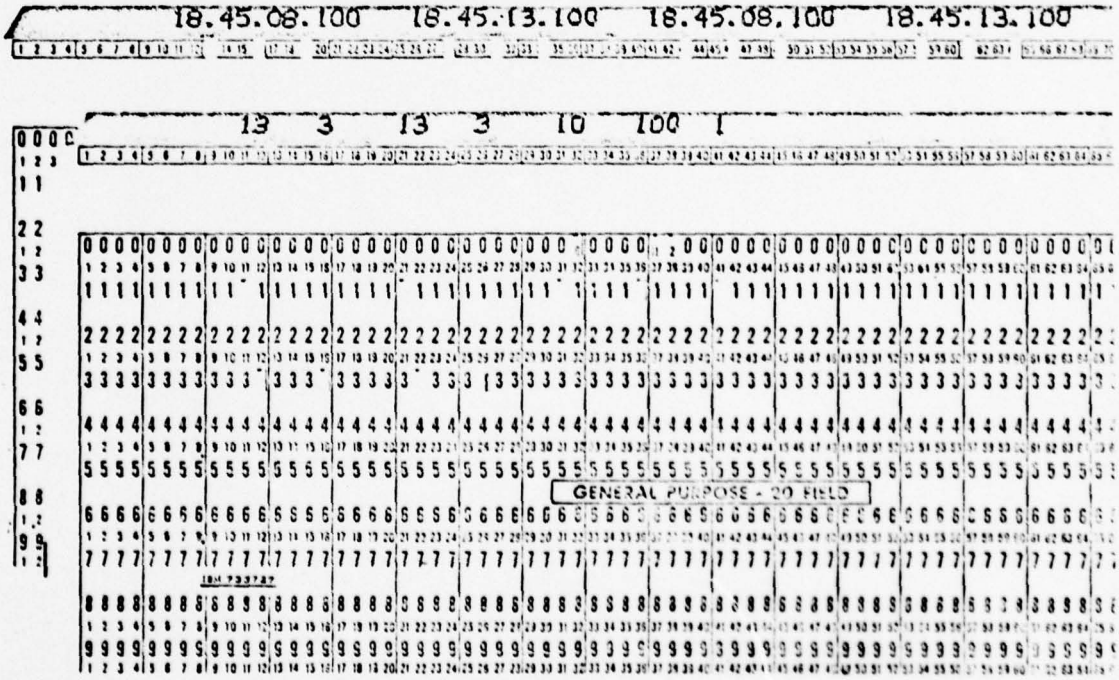
The parameter cards follow the transfer card. Each distinct correlation run requires two parameter cards. There is no limit to the number of runs which may be performed after the transfer card. A correlation run is defined to be the cross-correlation of two files of data as specified by the input parameters computed in the manner specified by these parameters. A series of correlation runs is terminated by an end card.

1. First Parameter Card: The first parameter card of a correlation run contains seven parameters. They are positive decimal numbers whose values do not exceed five digits. They are named: (1) REFFILE, (2) REFUNIT, (3) SIGFILE, (4) SIGUNIT, (5) QQ, (6) MM, and (7) HOPFLAG and their positions on the parameter card are fields 2 through 8 respectively, each parameter occupying one field. The most significant digit of each parameter must be punched in the left most digit of its field; otherwise the card input s/r will consider this an input error. WFA User's Manual gives a full description of card fields and their positions.

REFFILE is the file on mag tape on which the reference data is located; REFUNIT contains the unit number of this tape. SIGFILE contains the file number of the signal echo data on mag tape while SIGUNIT has the unit number of this tape. QQ is the number of segments into which the data is to be divided. It only has significance in the segmented mode. In continuous mode, this number is zero. MM is the value of the maximum relative time delay. Like QQ, it has meaning only in the segmented mode. HOPFLAG is the

switch indicator. For continuous cross-correlation, set HOPFLAG = 0; for segmented cross-correlation, set HOPFLAG = 1.

2. Second Parameter Card: The second parameter card of each correlation run contains the time code limits of the reference and echo files of the data to be selected. The four parameters on this card are: (1) initial reference time code, (2) final reference time code, (3) initial echo time code, and (4) final echo time code. Initial reference begins in field 2; final reference begins in field 5; initial echo time code begins in field 8 and the final echo time code begins in field 11. Each time code consists of twelve characters. There are two hours digits, two minutes digits, two seconds digits, and three milliseconds digits. Hours, minutes, seconds and milliseconds fields are separated by periods. The following time code example, 18.45.08.100, signifies 6:45 pm, 8-1/10 seconds. An example of these parameter cards is given below.



3. Core Option for Reference: To select the CORE option for reference as described above, delete tooth timecodes for the reference and punch the word CORE in Field 2.

C. END Cards

An END card terminates a sequence of correlation runs. It returns the program to MONITOR control.

III. PROGRAM OUTLINE: GENERAL DESCRIPTION

The executive routine CLIPCORR calls upon the following major subroutines to perform the task of the program: (1) CARDINPUT, (2) TRANTIME, (3) READPACK, (4) RDUNPACK, (5) COMPCORR, (6) PRINTOUT.

A. CARDINPUT

CARDINPUT reads in the two parameter cards, translates the information and stores the values.

B. TRANTIME

TRANTIME translates the echo and reference initial and final time code tables to milliseconds. Inputs are the nine word tables REFLIMLO, REFLIMHI, ECHLIMLO and ECHLIMHI. Outputs are the cells labeled MILIREFLO, MILIREFHI, MILIECHLO, and MILIECHHI.

C. READPACK

The subroutine READPACK reads in reference data from magnetic tape as specified by the time codes, clips it retaining only the sign information of each element and packs it into core memory 30 elements to the word. Inputs to READPACK are mag tape, FILE and UNIT information, and the reference time limits. Output is into the general core area and the table limits word, REFTAB.

D. RDUNPACK

This procedure reads and unpacks an echo signal from mag tape into the core area. Inputs to RDUNPACK are FILE and UNIT information,

mag tape and the echo time code limits. The data are read in, unpacked and sign extended from twelve bits to fifteen bits and stored in the output table AREA as half words. The word AREA contains the upper and lower limits of the AREA table.

E. COMPCORR

This routine is the heart of the CLIPREF system. It does all the housekeeping, updating and so on for the various correlation computations. It initializes and executes the subroutine, GENCORR, which performs cross-correlations as specified by inputs. Inputs to COMPCORR are such quantities as QQ, MM, HOPFLAG, PKFLAG, ECHNUM, and REFNUM. The output of COMPCORR is a number of cross-correlation data written on a scratch file of the program tape.

F. PRINTOUT

This routine prints the output of a correlation run. Input is from the program mag tape. Output is on the high speed printer. There are two outputs: (1) an output of data and index values and, (2) a graphical output of the data. The segmented mode of the graphical output prints out as many graphs as segments with the time codes of each segment on the header of the graph.

IV. CARDINPUT

A. Translation of First Card Parameters

The parameter card is first read into core using the CARD S/R in MONITOR. The parameter cells REFFILE, REFUNIT, ECHFILE, ECHUNIT, QQ, MM, and HOPFLAG are in order. They are cleared. The information in CARDBF, the input buffer used by CARD, is translated word by word from XS-3 codes to octal numbers and stored. Each parameter is assumed to be an unsigned decimal integer. Each parametric field is unpacked character by character into a nine word table called PACK.

The unpacking loop then tests the characters and stops when a blank or an illegal character is reached. An error condition is generated when either of the following conditions exist: (1) The first character of a field is blank or (2) an illegal character code is in the field, i.e., a non-numeric character other than a blank. If such an error is generated, the following message is typed on the flex: FIRST PARAMETER CARD ILLEGAL. The computer than 4-stops at the call to CARDINPUT in the EXEC. All the operator need do then is punch a correct card, stick it at the bottom of the hopper, and hit high speed. If all is well, the parameters are properly stored and the second parameter card sequence is initiated.

B. Translation of Second Card Parameters

Immediately after the second parameter card is read into core, the third word of CARDBF is checked to see if it contains the word CORE. If it does, the subroutine jumps to the translation of the echo time codes, skipping translation of the reference time codes. Each time code parameter is unpacked by the same loop. The loop first unpacks each parameter into PACK, then it transfers PACK to the appropriate 9 word table. These output tables are REFLIMLO, REFLIMHI, ECHLIMLO, and ECHLIMHI. As each parameter digit is unpacked, it is checked for numeric value. If any of the digits checked are non-numeric, the following message is printed on the flexowriter: SECOND PARAMETER CARD ILLEGAL. The program then stops at the call to CARDINPUT in the EXEC. The card may then be corrected by the operator; however, both cards must be read to restart the job. If all is well, after the parameter tables are filled, CARDINPUT transfers control to the EXEC.

C. Recognition of End Card; Exit Condition

If the first card read in by CARDINPUT is an end card, control is transferred back to the MONITOR.

V. TRANTIME

A. Specific Explanation

TRANTIME translates echo and reference initial and final time tables to milliseconds. A subroutine named PACKTIME is used to do this. The four input tables each containing nine time code digits, REFLIMLO, REFLIMHI, ECHLIMLO, and ECHLIMHI are translated to milliseconds and stored in cells MILIREFLO, MILIREFHI, MILIECHLO, and MILIECHHI respectively.

PACKTIME translates timeburst coded time to millisecond time. Input is the nine word TIMER table; output is the cell MILITIME.

B. Compute REFNUM and ECHNUM

TRANTIME computes REFNUM and ECHNUM by the following formulae:

REFNUM = (MILIREFHI - MILIREFLO)(25)/2

ECHNUM = (MILIECHHI - MILIECHLO)(25)/2

TRANTIME then returns control to the EXEC.

VI. READPACK

A. Specific Information

This procedure reads in reference data and packs it into core. The contents of REFFILE and REFUNIT are stored into FILE and UNIT and A is cleared in order to initialize POSTIME. POSTIME positions the specified tape unit to the specified cell, searches for the first time code of the data and moves it to TIMER. READPACK then executes PACKTIME and stores MILITIME into REFSTART. REFSTART is compared with MILIREFLO. If MILIREFLO is smaller than REFSTART,

the following error is printed onto the flexowriter: INITIAL TIMECODE NOT IN FILE. If all is well, the subroutine SHOVEMIN is initialized and executed. SHOVEMIN packs the reference data into core. After execution of it, the reference data table limits cell REFTAB, is filled and READPACK returns control to the exec.

B. POSTIME

POSTIME has two functions. On the first call of a data input sequence, it positions a tape, spaces past the header record, reads the first data record and transfers the first time burst to TIMER. On subsequent calls, it merely reads in an additional record of data. Inputs are the cells FILE and UNIT. Outputs are the tables TIMER and DIG.

The A-register is the switch for POSTIME. If A is zero, POSTIME assumes a first call. If A is non-zero, it merely reads another record. If on an initial call to POSTIME the formatted digitized data is searched and no time code is found, the following message is printed on the flexowriter: INITIAL TIMECODE MISSING. The job is then aborted with a core dump being printed on the high speed printer.

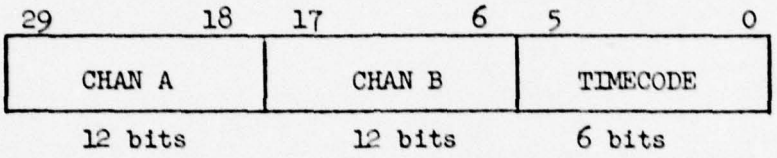
Input data from mag tape is read into table DIG. DIG is a 1200 decimal word table designed to use and handle input data only. If, when reading a record of data, TAPE reads an end of file, POSTIME executes an abnormal exit. If POSTIME is executed by READPACK, the following message is typed on the flexowriter: FIRST REFERENCE RECORD MISSING.

C. Mag Tape Input Data Format

A complete section on quantity or input data is called a file. Each file of data consists of a set of records. There is no

set number of records to a file; however, each record is exactly 1200 words long. The first record of each file is a header record and contains general information about the data to follow. Subsequent records contain the formatted digitized data itself. In the last record of a file, the end of the data is marked by the filling of the rest of the table with negative zeroes. Following this record is an end of file marker.

The data word format is as follows:



The upper 24 bits of the word are divided into two fields of 12 bits each called CHAN A and CHAN B. The lower six bits contain the time code information. There are two modes in which data can be packed into these records and stored on tape, dual mode and single mode. In dual mode, two separate records of data are stored, one in CHAN A and the other in CHAN B. In single mode storage, the information is stored alternately in CHAN A and CHAN B, two samples to the word with there being only one channel of data. All input data for CLIPREF is single mode data.

Every hundred samples, a 13 word time code burst is generated. In single mode format, this burst occurs every 50 words. The time code burst contains 13 binary coded decimal numbers. These numbers are in bits 0-3 in the TIMECODE field. Bit 4 is the time burst marker, and when lit, indicates the first word of a time burst. The first three codes of a burst are milliseconds; the next two are seconds; the two following are minutes digits; the next two are

hours. The next three are the run number; and the last digit is a code number. A more complete description of the data format is described in the TRG report, TRG 023-TN-66 entitled: Sea Trial Data Acquisition and Analysis Equipment.

Although a time code burst occurs every 50 samples throughout the input data, the input routines of CLIPREF translate only the first time code burst of each file of data and ignore all the rest.

D. Core Data Format

1. GENERAL Table Design: Library routines run from address 140 to 7000. The CLIPREF program itself starts at 7000. The MONGOOSE MONITOR runs from 70000 on up. CLIPREF with all its instructions and fixed length tables runs from 7000 to about 20500. This leaves a general data storage area from approximately 20500 to 70000. This area is called the GENERAL table, and it is here that reference and echo data are stored. Reference and echo data are not fixed length data, hence, they are stored in core in relative table areas.

Reference data is packed in first, the starting address being the starting address of GENERAL. When packing is completed, the upper and lower address limits of the table are stored in the cell labeled REFTAB, the upper limit in the upper half and the lower limit being stored in the lower half. The echo data is packed in core immediately above the reference data. After storage into core, its table limits are stored into AREA in the same manner.

2. REFTAB Table Design: The reference table is packed containing only sign information. The data are packed 30 bits to the word, starting from the left going to the right. A set bit

indicates a negative sign; a cleared bit indicates a positive sign. In order to perform any correlation at all, all the reference data must be read into core. Since the available cells number 20000, this limits the reference data input to about 600,000.

E. SHOVEMIN

This procedure packs one bit sign data from mag tape 30 bits to the word and stores it as specified by B1, which contains the address to store the packed value. Inputs are B1, BB5 and NN. BB5 contains the initial number of values to skip before starting the packing of data. NN contains the number of data points to pick up. Outputs are REFTAB and the table to which it refers.

The subroutine first spaces ahead the proper number of values, making use of POSTIME to do this. If POSTIME reads an end of file, an error is generated and the following message is typed on the flexowriter: INITIAL TIMECODE NOT IN FILE. After spacing forward to the correct starting point, SHOVEMIN starts packing the data and stores them in the reference table. As each data word is pulled out of table DIG, it is tested for a value of negative zero. If a negative zero is found, the following error is typed on the flexowriter: LIMITS EXCEED DATA ON FILE. If the length of the reference data exceeds the core limits (about 600,000), an error is generated and this message is typed on the flexowriter: CORE OVERFLOW. If all is well, after packing in the data, SHOVEMIN returns control to CLIPCORR and the system proceeds to read the echo data.

VII. RDUNPACK

This procedure reads and unpacks an echo signal from mag tape into the relative table AREA. Its operation is comparable to READPACK.

ECHFILE and ECHUNIT are stored into FILE and UNIT. The time code parameters are converted to milliseconds. PASSFORD is used to space to the beginning of the signal from the beginning of the file. The echo date is packed in a different format, but other than that, packing and storage is similar to READPACK.

A. PASSFORD

This procedure passes forward in order to reach the first cell of data. BB5 is used as the number of samples to skip. POSTIME is used to actually pass records forward.

B. AREA Table Design

The AREA table starts directly after the reference table and ends at address 67777. Each word of AREA contains two samples of data at most. The input data are signed 12-bit numbers. When packed into AREA, they are sign-extended to half words. Data is packed into AREA sequentially in the upper half of the word from the lower limit to as high as 67777. When the upper half words have been filled, then data packing starts in the lower half words from the lower limit to 67777. In the course of such packing, one of three conditions will arise: (1) data input will not completely fill the upper half words of the table, (2) data input will fill the upper half but not the lower half words of the table, (3) data will completely fill the table and not all the echo can be read into core. If (1) occurs, then the upper limit of AREA is set where it stopped and program is exited. If (2) occurs, then the upper limit of AREA is set to 67777 and subroutine is exited. If (3) occurs, then the upper limit of AREA is set to 67777, a flag is set carrying the information that not all the data could be read in, and RDUNPACK is exited.

C. Output Function Words

The following variables are also computed by RDUNPACK in the course of reading the echo signal data into core. They are subsequently used by COMPCORR to determine appropriate courses of action.

1. PKFLAG: PKFLAG is the packed echo signal flag. If PKFLAG = 0, then all echo data selected has been packed into core. PKFLAG = 1 indicates that not all signal data was packed into core.

2. DIGPT: DIGPT is the last input word referenced. It contains the index number of table DIG of the last signal datum stored into AREA. If, upon subsequent execution of COMPCORR, it is desired to read an additional amount of echo signal data, DIGPT indicates the point from which the last sample had been read.

3. DCTR: DCTR is the echo data counter. It contains the number of echo samples which have been read in a run regardless of the number actually present in core. DCTR is usually equal to NAR, but if AREA has been updated to input more data, it will contain the number of total data.

4. ICTR: ICTR is the initial data item counter. It contains the index number of the first echo sample currently in core. It is cleared by RDUNPACK, and remains zero until AREA is updated and subsequent data is read. e.g., suppose ICTR = 0 and AREA is updated by reading 300 more samples. The whole table must be slid down 300 samples with the bottom 300 deleted from the table. ICTR is then incremented by 300.

5. NAR: NAR is the number of AREA items. It contains the total number of echo signal samples presently in core. The following relationship holds among these last three quantities:

NAR + ICTR = DCTR

VIII. COMPCORR

The COMPCORR routine with its attendant subroutines constitutes the heart of the CLIPREF system. The other parts of the system perform merely input, output and housekeeping functions. COMPCORR does the actual cross-correlation computations.

Upon entry to COMPCORR, the program tape is positioned to the scratch file and the cross-correlation output table, CCOUT, is cleared. HOPFLAG is interrogated and control is given over to the appropriate mode, either continuous or segmented.

A. Continuous Mode

The definition and selection of continuous mode cross-correlation has already been discussed. Four distinct conditions can arise in continuous correlation: (1) All signal data are in core and the signal is longer than the reference, (2) all signal data are in core and the reference is longer than the signal, (3) not all signal data are in core and the signal is longer than the reference, and (4) not all signal data are in core and the reference is longer than the signal.

1. ECHNUM>REFNUM, PKFLAG = 0: This is probably the case which is used most often. All correlation is computed with one call to GENCORR with no updating. The correlation output values are computed by Eq(7).

2. ECHNUM>REFNUM, PKFLAG = 1: The correlation output values are still computed by Eq(7) in this case, but now the AREA table must be updated from time to time since the signal data are not all in.

3. REFNUM≥ECHNUM, PKFLAG = 0: All correlation values are computed in one call to GENCORR with no updating. Eq(6) is used to make the computations.

4. REFNUM≥ECHNUM, PKFLAG = 1: This condition is impossible because, no correlation can be computed by Eq(6) unless all echo data are in core. An error is generated and the following message is typed out: CORRELATION IMPOSSIBLE ECHO FILE TOO LONG.

B. Segmented Mode

Similarly, the definition and selection of the segmented mode has already been discussed. However, in this mode, the reference and the signal must be made to be the same length. Each segment to be correlated must satisfy the following condition: SEG≥2MM + 1. If this is not the case, then an error is generated and the following message is typed: MM TOO LARGE FOR SEGMENT LENGTH. Three cases can arise in segmented cross-correlation: (1) all data in core, (2) some data not in core, but at least one whole segment is in core, and (3) some data not in core and part of first segment not in core.

Case (1): Segmented cross-correlation performed without updating.

Case (2): Segmented cross-correlation performed with updating.

Case (3): The segment is too long to perform a cross-correlation. An error is generated and the following message is typed: CORRELATION IMPOSSIBLE, SEGMENT TOO LONG.

C. GTFLAG Modes

GTFLAG stands for greater than flag. It has the value of 0 if REFNUM≥ECHNUM and the value of 1 otherwise. GTFLAG is an input

of GENCORR and essentially tells it whether to j-index through the echo data or through the reference data. In continuous mode GTFLAG is 1 for cases 1 and 2 and 0 for cases 3 and 4. GTFLAG is always 0 for segmented mode cases.

D. PKFLAG Modes

PKFLAG indicates whether or not all the data have been read. If all echo data are in, correlations may always be made. If not, there are cases in which correlations are impossible. In continuous mode, this occurs when the reference is longer than the echo. In segmented mode, this occurs when the segment length is longer than the signal in core.

E. UPDATE S/R

This procedure updates table AREA by an amount equal to ELO and reads in more data as available. ELO is mainly an input for GENCORR and is the initial echo signal index. UPDATE is called only after GENCORR has been executed. When UPDATE is called ELO + VAL has exceeded NAR, the number of echo values in AREA. One of three conditions may occur when UPDATE is called: (1) ELO>NN, (2) ELO≤ NN, and (3) abort condition. After GENCORR has been executed, NN contains the number of whole words in the table AREA.

1. Case 1: ELO>NN: In this case ELO samples on the bottom will be deleted. Samples in the lower half of the table, starting with the ELOth sample will be moved to the beginning. When all samples in the table have been moved, the rest of the table is cleared and the input of additional data is initiated. Data input at this stage is similar to the data input processing of RDUNPACK.

2. Case 2: ELO≤NN: ELO number of samples at the bottom of the table are deleted. Samples in the upper half of the table move up ELO positions. At the bottom of the upper half, are inserted samples from the top of the lower half. Then the lower half table is updated. New samples are packed in at the bottom of the lower half.

In both cases 1 and 2, ICTR will be incremented by ELO. If all data is read in, PKFLAG will be cleared and NAR will be updated so as to correspond to the actual number of samples currently in AREA.

3. Abort Condition: An abort condition will occur in the data input stage of UPDATE under the same conditions that would cause an error generation in the data input of RDUNPACK. The same action will be taken as in RDUNPACK.

F. GENCORR

COMPCORR essentially does the housekeeping and initialization for the cross-correlation. GENCORR, the largest subroutine in CLIPREF, computes the correlation values and writes them on mag tape.

1. Indexing and Input Parameters: Inputs are RLO, ELO, VAL, NAR, JJ, and GTFLAG, Indices used are B1, B2, B3, BB4, and B5. RLO is the initial reference index. It contains the index number of the first datum to be pulled out of the reference table. ELO is the initial echo index and contains the index of first datum to be pulled out of the echo signal table. VAL is the segment of the correlated interval. It contains the number of correlation pairs that are to be compared. JJ is the correlation index shift. It contains the number of values and shifts to be computed in any call to GENCORR.

B1 is the reference input index. It contains the address of the word in REFTAB containing data to be correlated. B2 is the echo input index. It contains the address of a word in AREA to be correlated with the reference. B3 is the value counter index. It counts the number of pairs being correlated and is indexed through VAL. BB4 is the shift counter index. It counts the number of shifts of the data and is indexed through JJ. B5 is the output counter index. It counts the number of data computed, and when the output table, CCOUT, is filled, CCOUT is written on tape, it is cleared, and B5 is reinitiated.

In the initial housekeeping, NN is computed. NN is the number of samples in the upper half of AREA. It is a value which is used constantly in the indexing of the correlation. Next the denominator of the right sides of Eq(6) or Eq(7) is computed. Immediately after this computation of the absolute sum, which is done only once per call to GENCORR, the general indexing cycles begin.

The general indexing cycle, which must be done for each correlation shift, is rather complex. Reference data which may come from any bit of a 30 bit word must be matched against signal data which may come from an upper word or from a lower half word. Reference data may start anywhere in a 30 data input word and may end anywhere in a 30 data input word. A signal data segment may lie entirely in the upper half of AREA, it may lie entirely in the lower half of the table, or it may spill over, some data in the upper half and some in the lower half.

The actual correlation is done by two main cross-correlation sequences, each of which will correlate 30 elements at a time. One sequence correlates reference data with upper half signal data; the other correlates it with lower half signal data. The index matching is divided into six phases; three for the upper sequence and three for the lower sequence. The variables for these phases are $\phi 1$, $\phi 2$, $\phi 3$, $\phi 4$, $\phi 5$ and $\phi 6$.

2. Upper Half Table: The general indexing cycle must first determine whether signal values will be pulled from upper half only, lower half only, or from both. If the signal must be pulled from upper half only or from both halves, the upper half sequence must be initiated. The variables $\phi 1$, $\phi 2$, and $\phi 3$ must be computed for this sequence.

$\phi 1$ is the index number of the first reference value in a reference data word corresponding to RLO in the reference data. To compute $\phi 1$, divide RLO by 30 and put the remainder in $\phi 1$. The quotient is added to the lower half of REFTAB and stored in B1. Phase 1 picks up the initial fraction of reference data in the upper half sequence. Phase 2 correlates whole reference words, phase 3 picks up the final fractional reference word for correlation. $\phi 2$ contains the number of times phase 2 is repeated. Add $\phi 1$ to the number of upper half values and subtract 30; then divide by 30. The quotient is $\phi 2$; the remainder is $\phi 3$. $\phi 3$ contains the number of reference values to be picked in phase 3. After computation of these parameters, the indexing cycle computes the lower sequence parameters if signal data carry over to the lower half; otherwise control is transferred to phase 1 correlation. A special case occurs

when the reference data are less than 30 and all in the same word. $\phi 1$ is computed, but control is transferred directly to phase 3, bypassing phases 1 and 2.

3. Indexing for Lower Half Table: Indexing for lower half correlation sequence, which involves the computation of $\phi 4$, $\phi 5$, and $\phi 6$ is greatly similar to the indexing of the upper half sequence. $\phi 4$, $\phi 5$, and $\phi 6$ functionally correspond to $\phi 1$, $\phi 2$ and $\phi 3$ respectively.

4. Updating Sums: Consider Eq(6) and Eq(7). If correlation is computed by Eq(6), then the denominator need not be updated at the end of a correlation. At the end of each correlation, GENCORR checks GTFLAG. If GTFLAG = 0, we are using Eq(6) and we increment RLO and jump to the general index cycle. Otherwise we are using Eq(7) and must update the denominator. GENCORR simply subtracts the absolute value of the first value and adds the absolute value of the next new value, according to the following relation:

$$S_{i+1} = S_i - |E_{ELO}| + |E_{ELO+VAL}| \text{ where } S_i = \sum_{i = ELO}^{ELO-1+VAL} |E_i|$$

5. Method of Computation: Eq(6) and Eq (7) are actually too slow to be used in the computer for the actual correlation computation and have been given here for heuristic purposes. Consider the numerator of Eq (7):

$$\sum_{i=1}^{VAL} R_i E_{i+j}, \; j = 0, 1, \ldots, JJ-1 \qquad (10)$$

Each element of the sum must be either plus or minus $|E_{i+j}|$. We define the following functions:

$$E_{i+j(+)} = \begin{cases} E_{i+j} \text{ when } R_i E_{i+j} \text{ is positive} \\ 0 \text{ otherwise} \end{cases}$$

$$E_{i+j(-)} = \begin{cases} -E_{i+j} \text{ when } R_i E_{i+j} \text{ is negative} \\ 0 \text{ otherwise} \end{cases}$$

Accordingly (1) becomes:

$$\sum_{i=1}^{VAL} R_i E_{i+j} = \sum_{i=1}^{VAL} E_{i+j(+)} - \sum_{i=1}^{VAL} E_{i+j(-)} \tag{11}$$

We also have the following relation for the denominator of Eq (7):

$$S_j = \sum_{i=1}^{VAL} |E_{i+j}| = \sum_{i=1}^{VAL} E_{i+j(+)} + \sum_{i=1}^{VAL} E_{i+j(-)}$$

$$\sum_{i=1}^{VAL} E_{i+j(+)} = S_j - \sum_{i=1}^{VAL} E_{i+j(-)} \tag{12}$$

By substitution, Eq (7) becomes:

$$\sigma(j) = \frac{\sum_{i=1}^{VAL} E_{i+j(+)} - \sum_{i=1}^{VAL} E_{i+j(-)}}{S_j} = \frac{S_j - 2\sum_{i=1}^{VAL} E_{i+j(-)}}{S_j} \tag{13}$$

A similar relation holds for Eq (6):

$$\sigma(j) = \frac{S_o - 2\sum_{i=1}^{VAL} E_{i(-)}}{S_o}, \quad j = 0, 1, \ldots, JJ-1 \tag{14}$$

Where

$$E_i(-) = \begin{cases} 0 \text{ if } R_{i+j} E_i \text{ positive} \\ -E_i \text{ otherwise} \end{cases}$$

and where

$$S_o = \sum_{i=1}^{VAL} |E_i|$$

Eq(13) and Eq(14) are the equations actually used in the computation of the correlation values. The matching of data pairs thus involves only the computation of:

$$\sum_{i=1}^{VAL} E_{i+j(-)} \text{ or } \sum_{i=1}^{VAL} E_{i(-)}$$

6. Output Format: The computed correlation values are stored in a whole word in CCOUT scaled up 28 bits.

IX. PRINTOUT

PRINTOUT is the output subroutine. It reads in from mag tape and prints it out as previously explained. The output is in two parts, the correlation value print-out and the cross-correlation graph print-out.

A. Correlation Value Print-Out

For this output the program generates a page feed, prints the title "clipped reference cross-correlation", then the date below it, a table header, and lastly, the correlation values to five decimal places in order as they appear on tape.

B. Cross-Correlation Graph Print-Out

For this output PRINTOUT generates a page feed, prints the title header and the date below. All the input parameters follow. Included is a three line explanation of the graph. Lastly follows the graph itself.

1. Key 1 Option: This output can be deleted by setting Key 1.

2. Continuous Mode Graph: The continuous mode graph's index starts at zero.

3. Segmented Mode Graphs: Segmented mode graph may have more than one segment. In each segment, the index starts at -MM.

The time interval for echo and reference is divided into QQ equal intervals. The time interval for each such segment is printed at the top of each segment's graph.

C. UNPACKTIME

UNPACKTIME is a subroutine used by PRINTOUT to set up the output of the time codes. It translates millisecond time to timeburst coded time. Input is MILITIME and output is the TIMER table.

D. SETEMUP

SETEMUP is a subroutine used by PRINTOUT to set up the output of the time codes. It translates the time codes as stored in the TIMER table and generates a string of literals for output on the printer in conjunction with the PRINT package. Input is TIMER; output is STRING2.

E. GRAPH

This procedure is used by PRINTOUT to print a cross-correlation graph. Inputs are CCOUT and PONUM. PONUM is the number of lines to output. GRAPH sets up horizontal lines every tenth column on every line. Every tenth line, vertical bars are set in the printer output table, GRAF. GRAPH uses the subroutines, SENDWO to send a line.

1. Output Format: 101 columns starting at column 19 make up a line. Correlation values are indicated by an asterisk. All output values are rounded off to the nearest 0.02.

2. SUBGRAF: This procedure stores a print out value for output in table GRAF.

3. SENDWO: This procedure sends the printer 19 blanks, then it unpacks the codes in GRAF and sends them to the printer. After this, it prints a line without a line feed.

F. PRTDATE

This procedure is used by PRINTOUT to print the current date.

G. SETTIME

This procedure computes a time code and sets up a string of literals to print it out. Inputs are REFLIML0 and B1. Output is STRING2; subroutines used are UNPACKTIME and SETEMUP.

X. TIMING

A. Data Input

Translation and read-in of parameter cards takes less than two seconds. The reading and packing time of input data is usually a matter taking less than 20 to 30 seconds. If the data is a long ways down the tape, the tape may take awhile to wind down to it, but usually input data time considerations are trivial.

B. Cross-Correlation Time Estimation

The time for cross-correlation depends upon four main factors: (1) the average indexing time, (2) the average time for matching one pair of samples, (3) the length of the correlation segment, and (4) the number of correlation output values.

The general index cycle time varies from a low value of 839.2 microseconds to 1904.8 microseconds, depending upon the path taken and GTFLAG mode. The average index time cycle for GTFLAG = 0 is 1267.9 microseconds whereas indexing the j-shift through the echo signal takes an average of 1475.9 microseconds, about 208 microseconds longer. Let I be the average indexing time, which we will take as 1475.9 microseconds.

Let τ be the average time for matching one pair of samples, S be the number of correlation pairs, and J be the number of correlation output values. The time to compute one output value would be $S\tau+I$, hence the total time, t, would be $t = J(S\tau+I)$.

The numerical value for τ is $\tau = 52.4$ microseconds. For S small, the indexing time I is a large percentage of the time required for correlation. As S gets larger, the effect of I on the correlation time gets smaller and smaller. Consider the following table:

Sample Size	Time (usec)	% Indexing
10	2000	73.7%
30	3048	48.4%
100	6716	22.0%
300	17196	8.57%
1000	53876	2.74%
3000	158,676	0.93%
10000	525,476	0.28%
30000	1,573,476	0.09%

C. CC Value Print-Out Time

To print one line of the ten value per line output takes about 0.250 sec/line $\pm$2%.

D. Cross-Correlation Graph Print-Out Time

To print one line of the graph takes about 0.0945 sec/line $\pm$2%.

SUMMARY

The CLIPREF system was based on the experience gathered from the production of the CLIPCORR programming system. In the CLIPREF system, it was important to speed up the computations; This was

done at the expense of considerable core. Computation elements which would have to be repeated millions of times in a correlation run were strung out thirty times per loop instead of being repeated merely once per loop. The result was highly successful in terms of correlation computation speed. The value of T was reduced from about 80 microseconds to 52.4 microseconds.

Additional information is included in the appendices. Appendix A is a list of possible error printouts and their significance. Appendix B is the flow charts of the main subroutines of the system. Appendix C contains examples of the outputs. Both outputs to two continuous correlation runs are shown. The first graph shows the data points unconnected. In the second graph we have connected the data points to show the sinusoidal nature of the data which we were correlating.

REFERENCES

1. TRG 023-TN-66, "Sea Trial Data Acquisition and Analysis Equipment."

2. WFA Area Computer Center User's Manual

3. Introduction to Sonar Technology, NAVSHIPS 0967-029-3010.

APPENDIX A

Error Print-Outs

1. FIRST PARAMETER CARD ILLEGAL - Illegal format on first parameter card. Program will 4-STOP at the beginning of the card input sequence. Correct the parameter card and place it at the beginning of the deck. Restart the program by hitting high speed.

2. SECOND PARAMETER CARD ILLEGAL - Illegal format on second parameter card. Program will 4-STOP at the beginning of the card input sequence. Correct the parameter card and place it after the first card at the beginning of the deck.

3. INITIAL TIMECODE MISSING - Indicates that no timecode has been found on the first record of data read into core. This usually indicates a bad record of data on mag tape. Program will give a core dump and terminate the job.

4. CORE OVERFLOW - Indicates that the amount of reference data has exceeded the general area core limits assigned to it. Program will give a core dump and terminate the job.

5. LIMITS EXCEED DATA ON FILE - Indicates that one of the parametric upper timecode limits of the input data is greater than the last timecode of the corresponding data file on mag tape. Therefore, that data cannot be read and unpacked into core. The program will jump to 0 and Fault.

6. FIRST REFERENCE RECORD MISSING - Indicates that the lower reference timecode parameter limit cannot be found on the reference input tape. The program will jump to 0 and Fault.

7. INITIAL TIMECODE NOT IN FILE - Indicates that the initial reference timecode is less than the first timecode on the reference data tape, and hence the timecode is not on the file. Program will Fault and jump to 0.

8. FIRST SIGNAL RECORD MISSING - Indicates that the lower signal timecode exceeds the last timecode on the signal input tape. Similar to FIRST REFERENCE RECORD MISSING. Program will jump to 0 and Fault.

9. CORRELATION EXCEED DATA AVAILABLE - Indicates that an error has been made in a call to GENCORR. The number of correlations called for has exceeded the amount of signal data in core necessary to make the computations.

10. CORRELATION IMPOSSIBLE, ECHO FILE TOO LONG - This error occurs when the reference is longer than the echo data and all the echo data cannot be read into core because of lack of core storage. Thus the correlation cannot be performed because the echo file is too long. The program jumps to 0 and Faults.

11. MM TOO LONG FOR SEGMENT LENGTH - This error print-out can occur only when the segmented correlation mode has been selected. It indicates that MM, the number of correlation shifts is larger than half the segment length. Correlation therefore cannot be performed. Program jumps to 0 and Faults.

12. CORRELATION IMPOSSIBLE, SEGMENT TOO LONG - This error can occur only when the segmented correlation mode has been selected. It indicates that the signal segment is too long for all of it to be read into core. The program ignores the run causing this error, jumps to the beginning of the executive routine and begins executing the next correlation run request.

CLIPPED REFERENCE CROSS-CORRELATION

ACT 10 07

INDEX	0	1	2	3	4	5	6	7	8	9
0	0.09684	-0.31839	0.03591	0.31491	-0.16167	-0.25103	0.20977	0.14767	-0.33102	-0.00998
10	0.34644	-0.12858	-0.24749	0.25358	0.20197	-0.33857	-0.08253	0.37391	-0.08621	-0.34220
20	0.23139	0.25882	-0.33571	-0.11879	0.39201	-0.03750	-0.37771	0.19788	0.30824	-0.32317
30	-0.17628	0.40123	0.01609	-0.40701	0.15647	0.35424	-0.30010	-0.23160	0.39880	0.07120
40	-0.42698	0.10909	0.38973	-0.26710	-0.27931	0.38450	0.12405	-0.43228	0.05824	0.41285
50	-0.22500	-0.31753	0.35810	0.17501	-0.42278	0.00640	0.42128	-0.17777	-0.34385	0.32004
60	0.21469	-0.39923	-0.04322	0.41344	-0.12550	-0.35649	0.27203	0.24524	-0.36287	-0.08801
70	0.39478	-0.07247	-0.35459	0.22054	0.26802	-0.31581	-0.12447	0.36075	-0.02233	-0.33858
80	0.10403	0.26872	-0.26117	-0.15120	0.31573	0.02239	-0.30990	0.10780	0.26082	-0.20210
90	-0.10655	0.26269	0.03918	-0.27064	0.05523	0.24139	-0.14233	-0.17082	0.20922	0.08654
100	-0.22438	0.00923	0.21311	-0.08207	-0.16570	0.14799	0.10341	-0.17444	-0.02784	0.17799
110	-0.03689	-0.15057	0.09305	0.11032	-0.12879	-0.05469	0.13962	0.00534	-0.12858	0.04357
120	0.10807	-0.07503	-0.07176	0.04704	0.03705	-0.10210	0.00000	0.00000	0.00000	0.00000

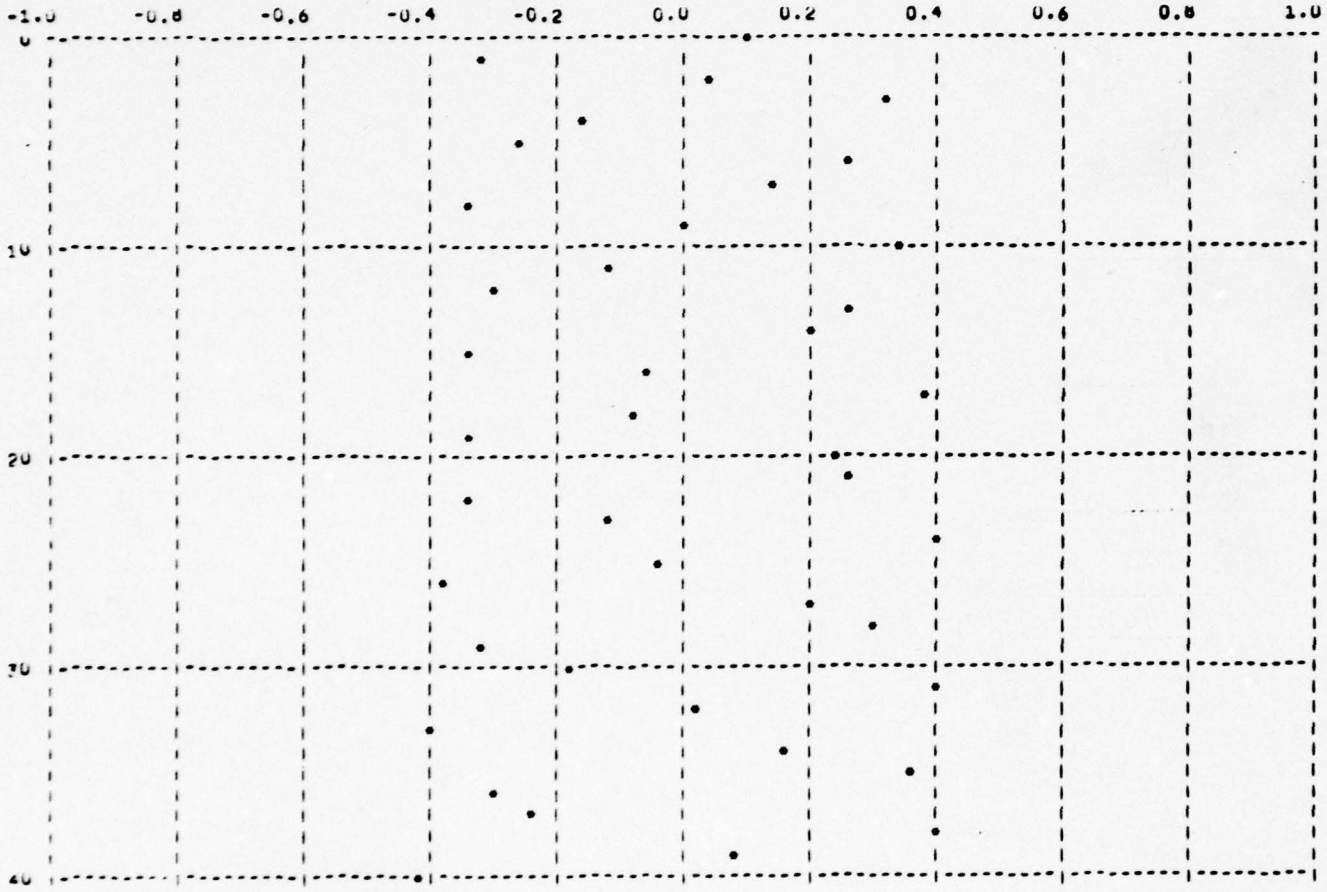
CLIPPED REFERENCE CROSS-CORRELATION

OCT 10 67

ECHOFILE= 12 REFFILE= 1
ECHOTAPE= 4 REFTAPE= 4
QQ= 0 MM= 0
CONTINUOUS OPTION

THE FOLLOWING IS A GRAPH OF CLIPPED REFERENCE CROSS-CORRELATION VALUES VS INDEX NUMBER
THE CLIPPED REFERENCE CROSS-CORRELATION VALUES ARE PLOTTED HORIZONTALLY FROM LEFT TO RIGHT
INDEX VALUES ARE PLOTTED FROM TOP TO BOTTOM

TIMECODE LIMITS=	INITIAL TIME	FINAL TIME
ECHO=	18.42.26.800	18.42.59.100
REFERENCE=	18.42.26.744	18.42.57.054



CLIPPED REFERENCE CROSS-CORRELATION

OCT 11 67

INDEX	0	1	2	3	4	5	6	7	8	9
0	0.61215	0.33768	-0.58950	-0.38546	0.57384	0.44380	-0.54120	-0.48839	0.51558	0.54251
10	-0.47341	-0.58219	0.43847	0.63076	-0.38755	-0.66390	0.34438	0.70480	-0.28598	-0.72995
20	0.23591	0.76182	-0.17166	-0.77723	0.11653	0.79868	-0.04855	-0.80319	-0.00917	0.81328
30	0.07840	-0.80640	-0.13644	0.80460	0.20473	-0.78612	-0.26095	0.77295	0.32627	-0.74301
40	-0.37848	0.71870	0.43874	-0.67855	-0.48523	0.64449	0.53874	-0.59543	-0.57791	0.55313
50	0.62305	-0.49656	-0.65355	0.44814	0.68951	-0.38675	-0.70999	0.33391	0.73535	-0.26980
60	-0.74503	0.21524	0.75935	-0.15076	-0.75813	0.09673	0.76153	-0.03396	-0.74976	-0.01775
70	0.74279	0.07725	-0.72122	-0.12512	0.70484	0.17999	-0.67458	-0.22275	0.65005	0.27181
80	-0.61254	-0.30814	0.58138	0.35045	-0.53823	-0.37973	0.50225	0.41413	-0.45542	-0.43519
90	0.41641	0.46109	-0.36752	-0.47377	0.32722	0.49119	-0.27805	-0.49539	0.23805	0.50440
100	-0.18998	-0.50100	0.15147	0.50259	-0.10557	-0.49221	0.06954	0.48721	-0.02666	-0.47084
110	-0.00620	0.46018	0.04538	-0.43879	-0.07465	0.42347	0.10988	-0.39813	-0.13483	0.37923
120	0.16540	-0.35043	-0.18578	0.32939	0.21162	-0.29912	-0.22747	0.27591	0.24873	-0.24430
130	-0.26012	0.22002	0.27699	-0.18795	-0.28415	0.16302	0.29676	-0.13064	-0.29995	0.10549
140	0.30851	-0.07318	-0.30796	0.04803	0.31268	-0.01594	-0.30853	-0.00897	0.30958	0.04043
150	-0.30200	-0.06494	0.29954	0.09558	-0.28864	-0.11911	0.28283	0.14083	-0.26879	-0.17123
160	0.25977	0.19945	-0.24279	-0.22044	0.23056	0.24689	-0.21056	-0.26610	0.19524	0.29029
170	-0.17221	-0.30699	0.15382	0.32829	-0.12765	-0.34201	0.10636	0.36015	-0.07747	-0.37060
180	0.05324	0.38503	-0.02196	-0.39185	-0.00456	0.40226	0.03764	-0.40505	-0.06581	0.41112
190	0.10019	-0.40955	-0.12940	0.41092	0.16433	-0.40455	-0.19390	0.40101	0.22849	-0.38950
200	-0.25741	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

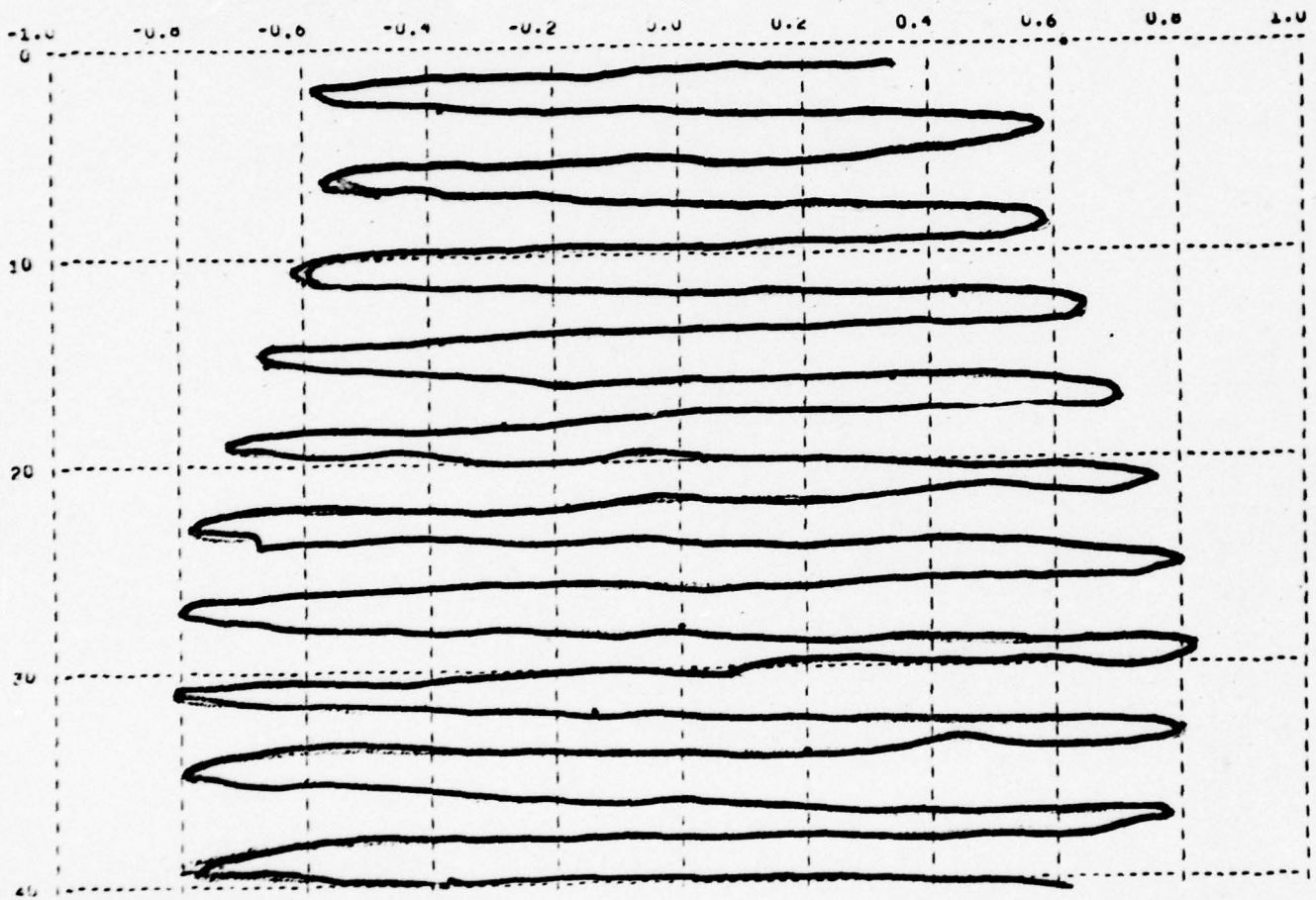
CLIPPED REFERENCE CROSS-CORRELATION

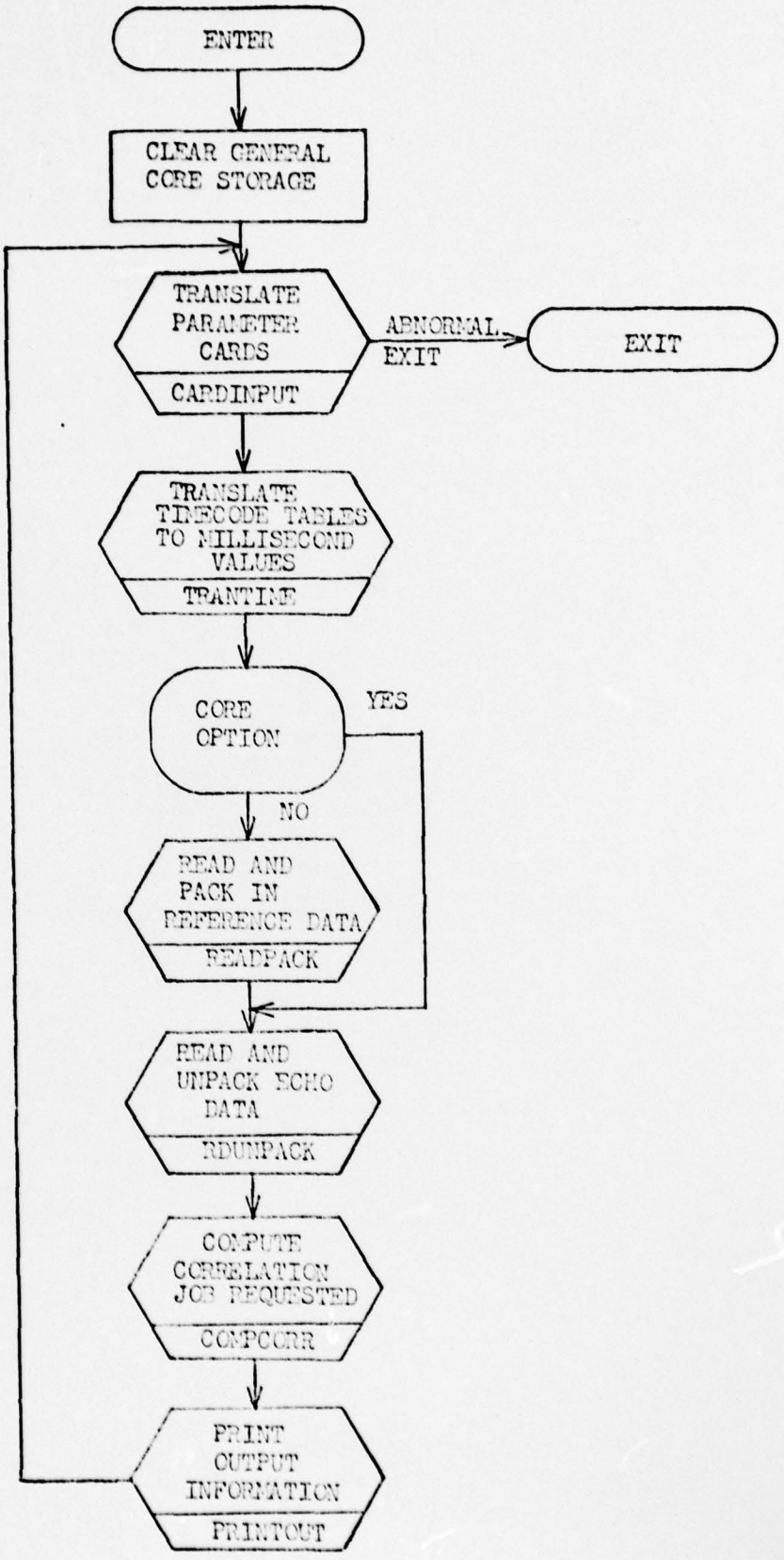
OCT 11 07

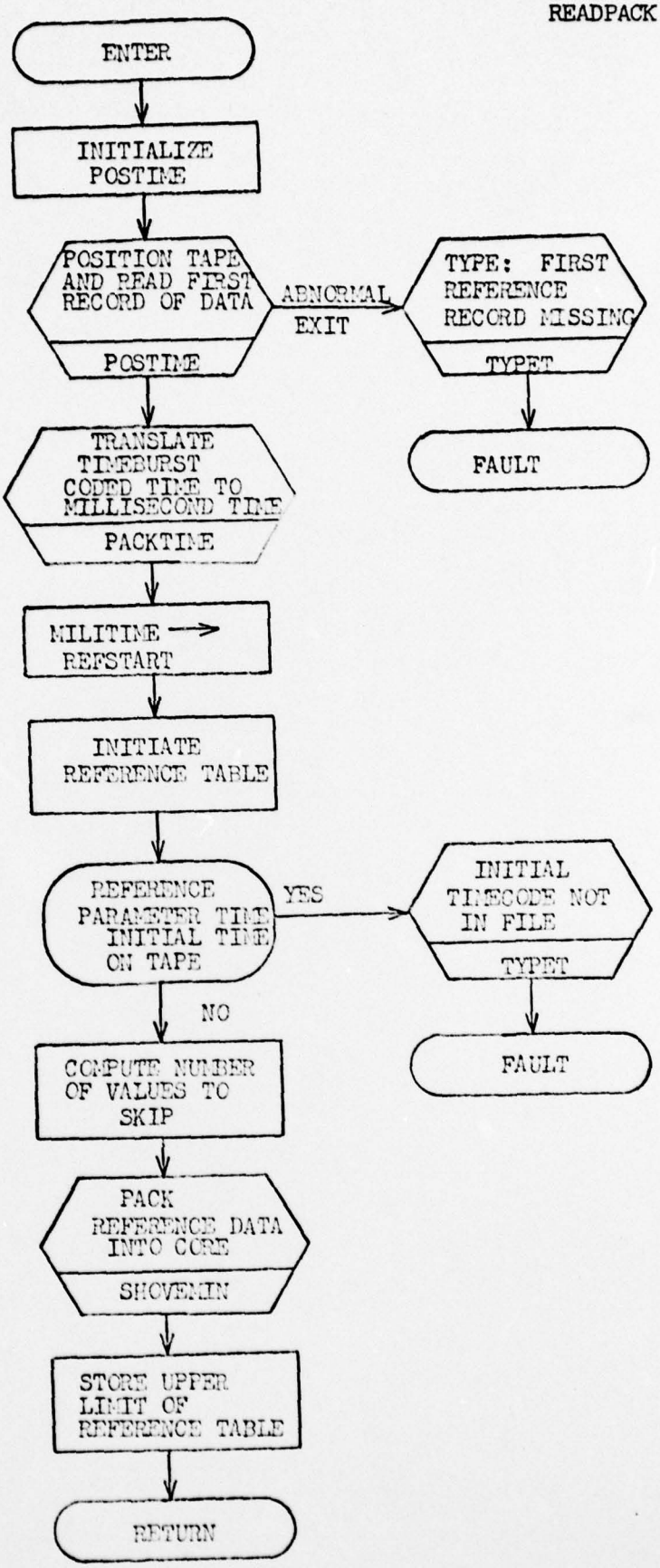
ECHOFILE= 62 REFFILE= 10
ECHOTAPE= 4 REFTAPE= 4
OD= 0 AD= 0
CONTINUOUS OPTION

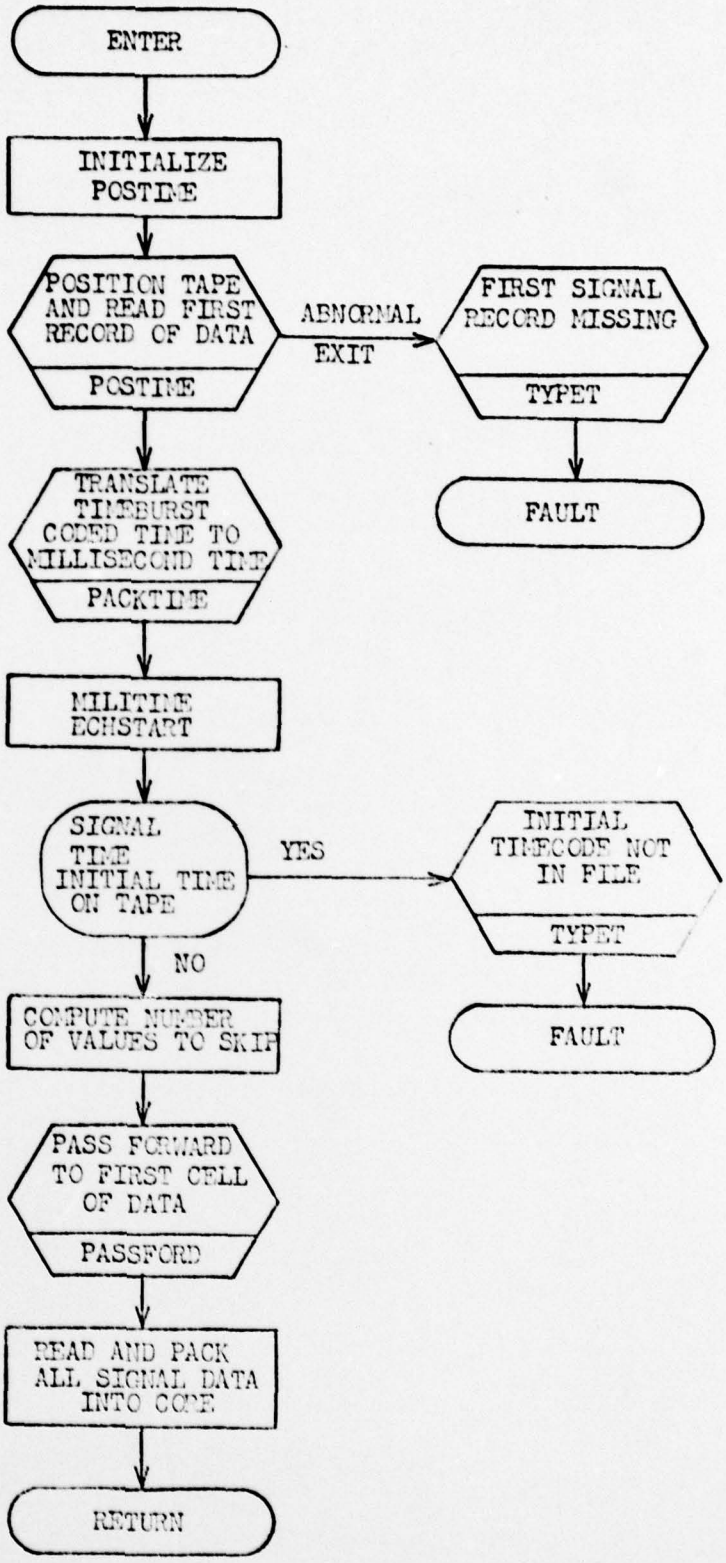
THE FOLLOWING IS A GRAPH OF CLIPPED REFERENCE CROSS-CORRELATION VALUES VS INDEX NUMBER
THE CLIPPED REFERENCE CROSS-CORRELATION VALUES ARE PLOTTED HORIZONTALLY FROM LEFT TO RIGHT
INDEX VALUES ARE PLOTTED FROM TOP TO BOTTOM

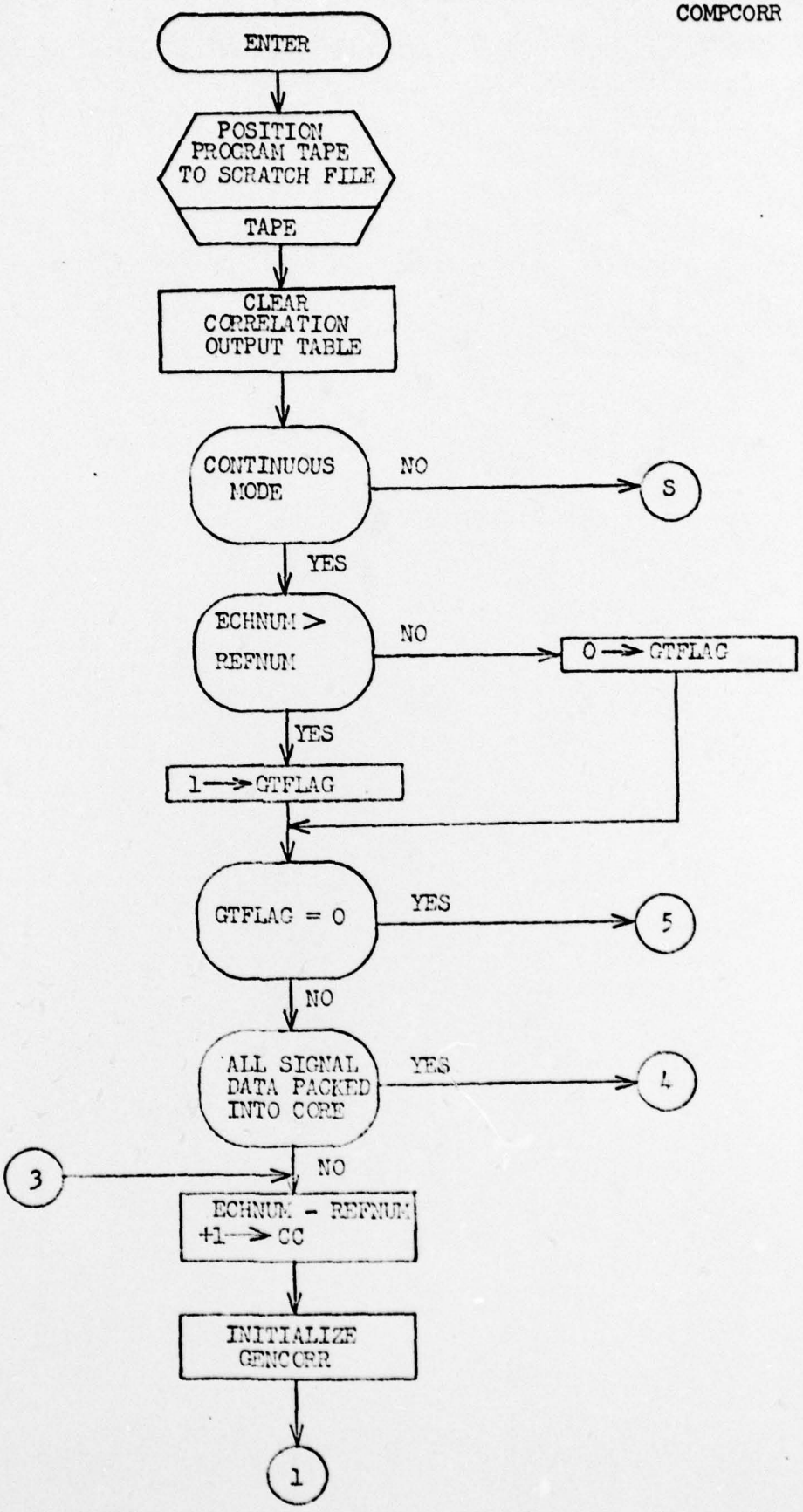
TIMECODE LIMITS=	INITIAL TIME	FINAL TIME
ECHO=	16.59.13.742	16.59.13.862
REFERENCE=	16.59.11.000	16.59.11.238

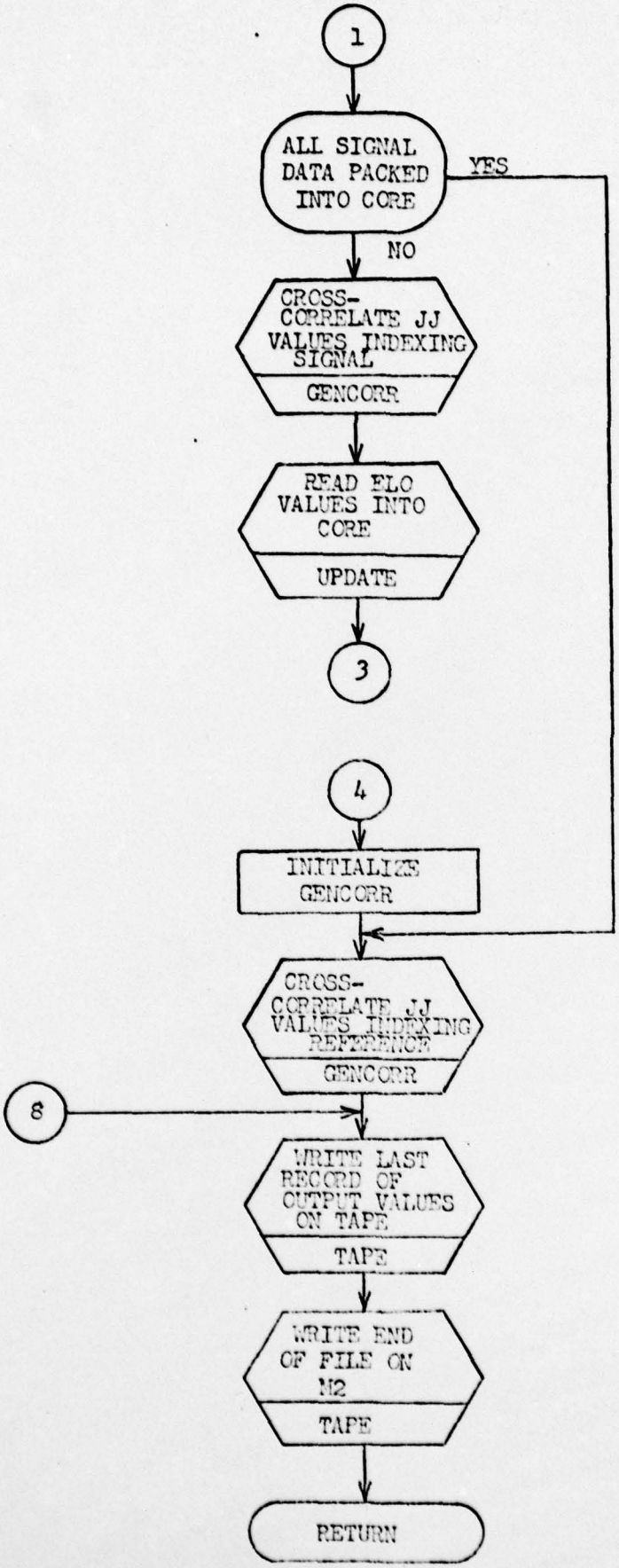


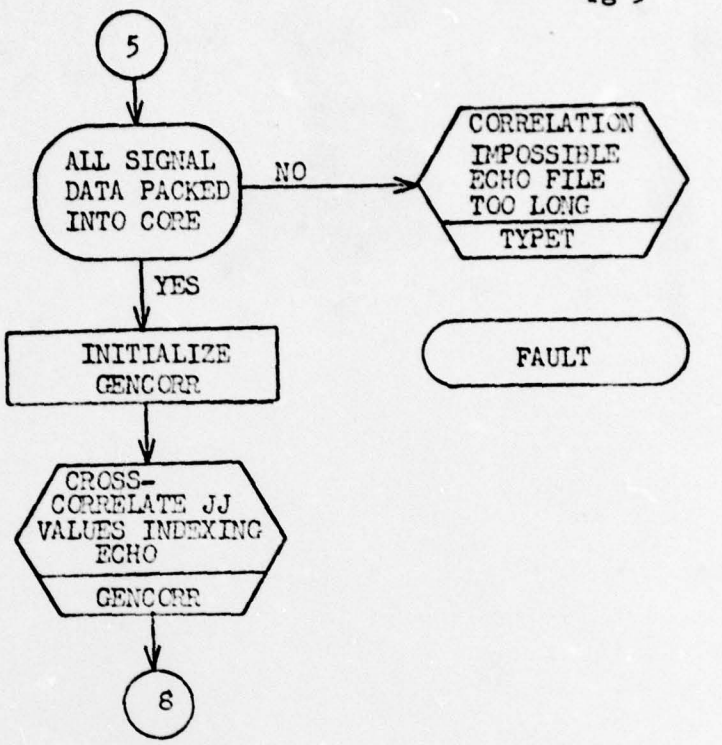

ENTER
CLEAR GENERAL CORE STORAGE
TRANSLATE PARAMETER CARDS
CARDINPUT
ABNORMAL EXIT
EXIT
TRANSLATE TIMECODE TABLES TO MILLISECOND VALUES
TRANTIME
CORE OPTION
YES
NO
READ AND PACK IN REFERENCE DATA
READPACK
READ AND UNPACK ECHO DATA
RDUNPACK
COMPUTE CORRELATION JOB REQUESTED
COMPCORR
PRINT OUTPUT INFORMATION
PRINTOUT

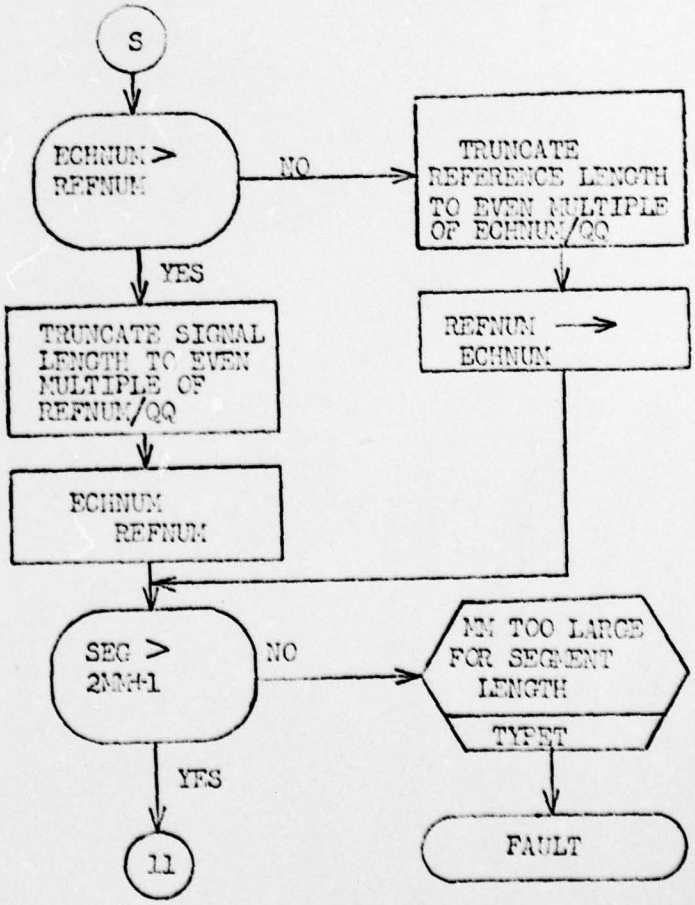

ENTER
INITIALIZE POSTIME
POSITION TAPE AND READ FIRST RECORD OF DATA
POSTIME
ABNORMAL EXIT
TYPE: FIRST REFERENCE RECORD MISSING
TYPET
FAULT
TRANSLATE TIMEBURST CODED TIME TO MILLISECOND TIME
PACKTIME
MILITIME → REFSTART
INITIATE REFERENCE TABLE
REFERENCE PARAMETER TIME = INITIAL TIME ON TAPE
YES
INITIAL TIMECODE NOT IN FILE
TYPET
FAULT
NO
COMPUTE NUMBER OF VALUES TO SKIP
PACK REFERENCE DATA INTO CORE
SHOVEMIN
STORE UPPER LIMIT OF REFERENCE TABLE
RETURN

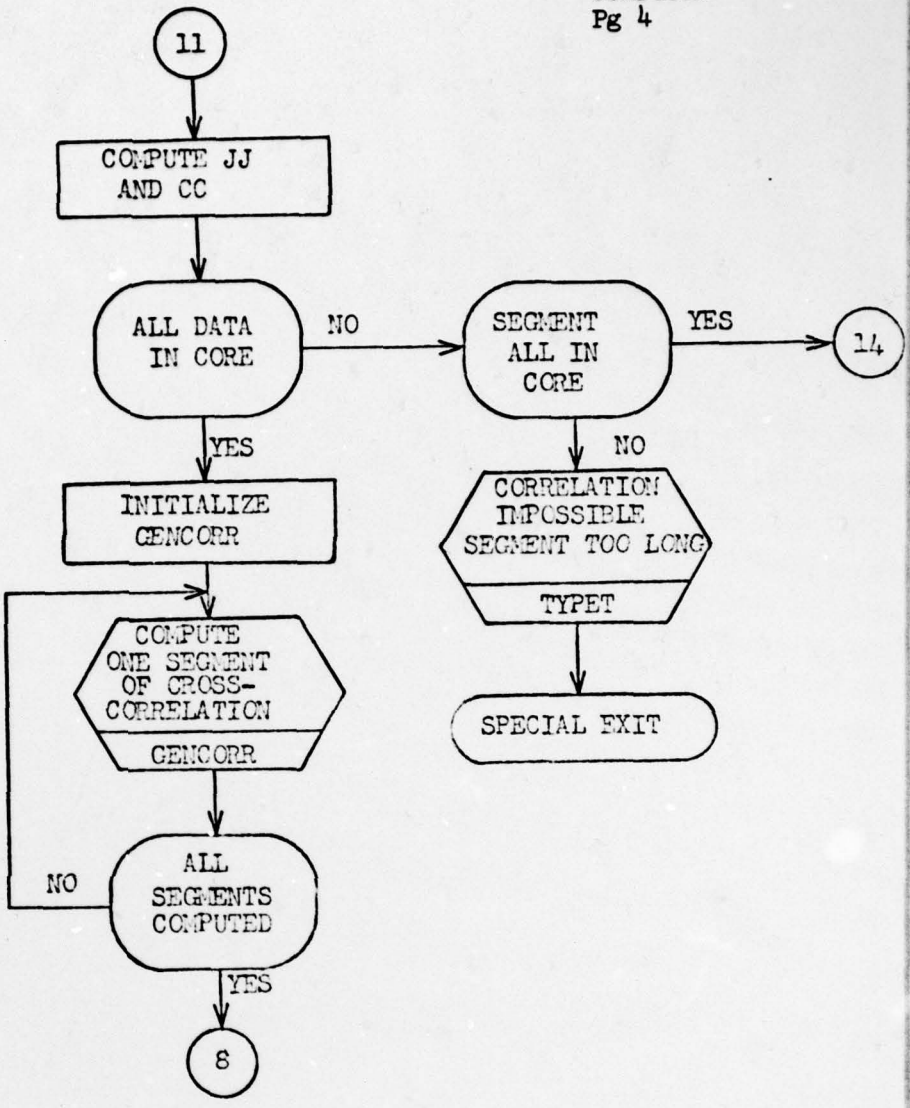

ENTER
INITIALIZE POSTIME
POSITION TAPE AND READ FIRST RECORD OF DATA
POSTIME
ABNORMAL EXIT
FIRST SIGNAL RECORD MISSING
TYPET
FAULT
TRANSLATE TIMEBURST CODED TIME TO MILLISECOND TIME
PACKTIME
MILITIME ECHSTART
SIGNAL TIME INITIAL TIME ON TAPE
YES
INITIAL TIMECODE NOT IN FILE
TYPET
FAULT
NO
COMPUTE NUMBER OF VALUES TO SKIP
PASS FORWARD TO FIRST CELL OF DATA
PASSFORD
READ AND PACK ALL SIGNAL DATA INTO CORE
RETURN

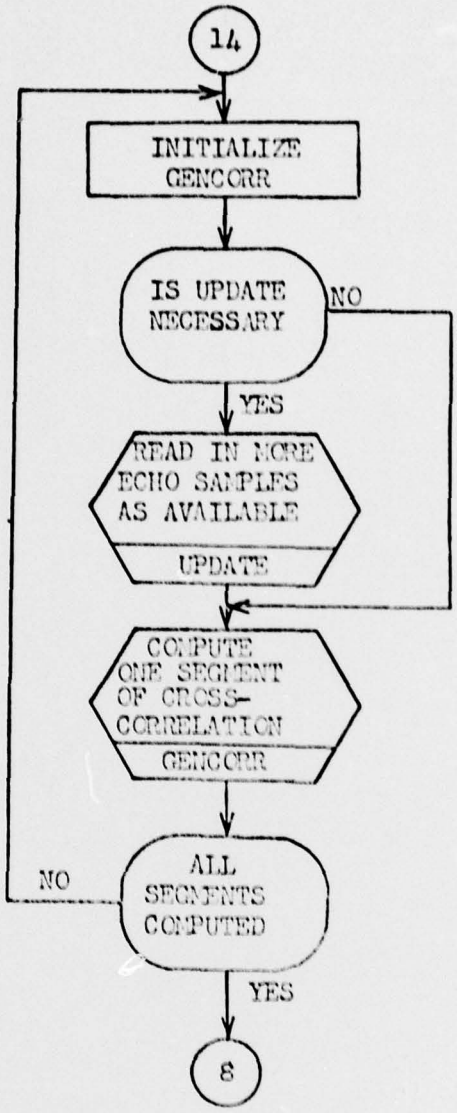

ENTER
POSITION PROGRAM TAPE TO SCRATCH FILE
TAPE
CLEAR CORRELATION OUTPUT TABLE
CONTINUOUS MODE
NO
5
YES
ECHNUM > REFNUM
NO
0 → GTFLAG
YES
1 → GTFLAG
GTFLAG = 0
YES
5
NO
ALL SIGNAL DATA PACKED INTO CORE
YES
4
3
NO
ECHNUM - REFNUM +1 → CC
INITIALIZE GENCORR
1

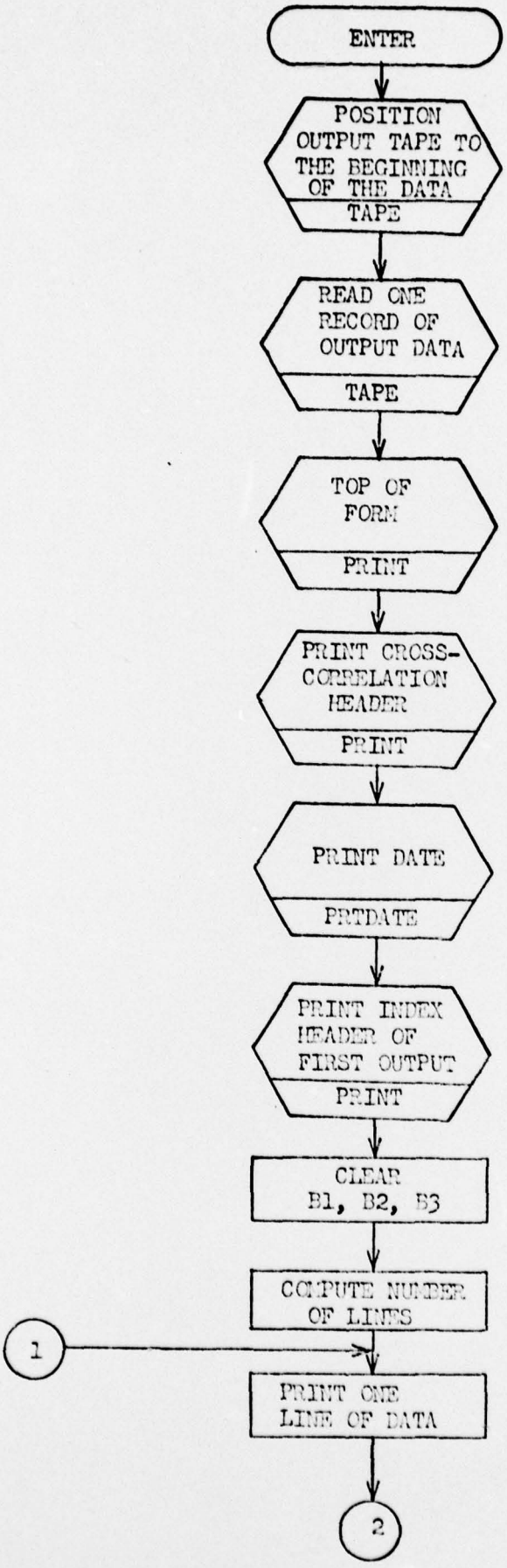

1
ALL SIGNAL DATA PACKED INTO CORE
YES
NO
CROSS-CORRELATE JJ VALUES INDEXING SIGNAL
GENCORR
READ FLO VALUES INTO CORE
UPDATE
3
4
INITIALIZE GENCORR
CROSS-CORRELATE JJ VALUES INDEXING REFERENCE
GENCORR
8
WRITE LAST RECORD OF OUTPUT VALUES ON TAPE
TAPE
WRITE END OF FILE ON M2
TAPE
RETURN

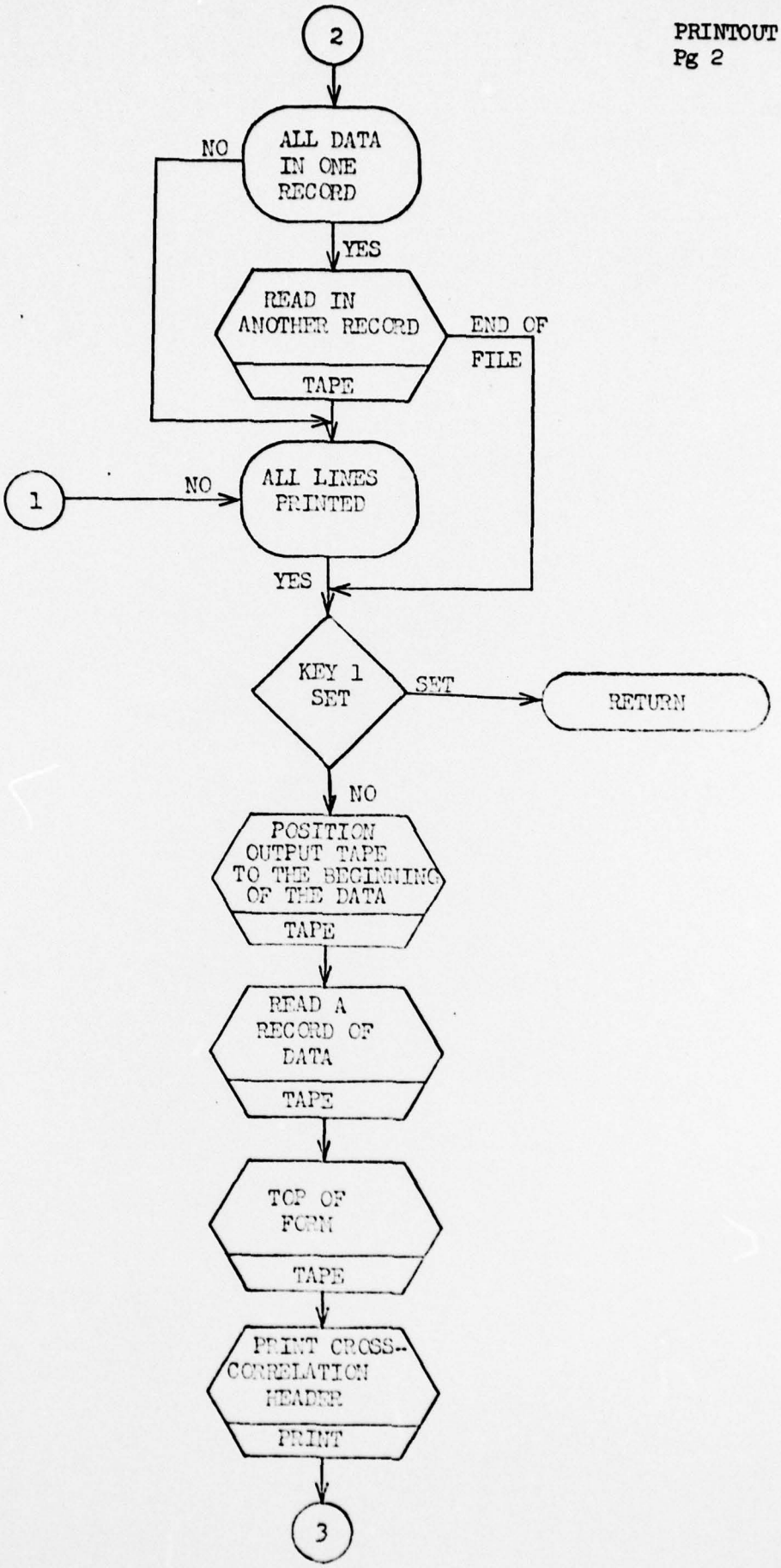

5
ALL SIGNAL DATA PACKED INTO CORE
NO
CORRELATION IMPOSSIBLE ECHO FILE TOO LONG
TYPET
YES
INITIALIZE GENCORR
FAULT
CROSS-CORRELATE JJ VALUES INDEXING ECHO
GENCORR
6

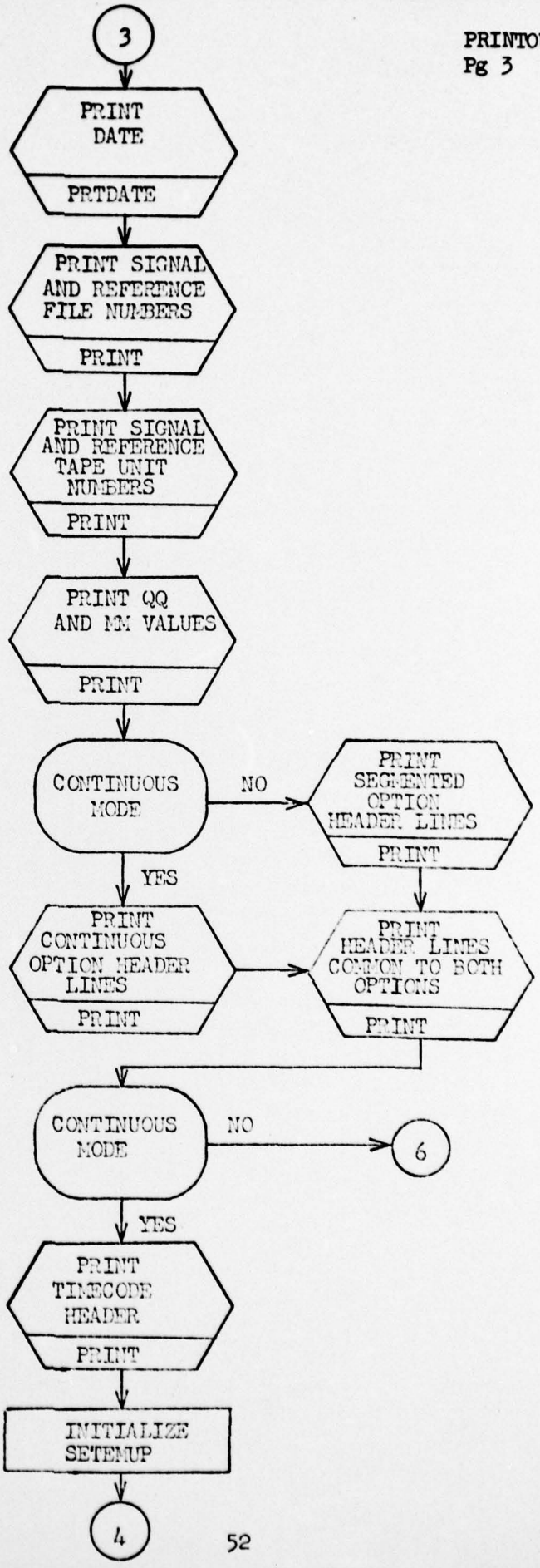

8
ECHNUM > REFNUM
NO
TRUNCATE REFERENCE LENGTH TO EVEN MULTIPLE OF ECHNUM/QQ
YES
TRUNCATE SIGNAL LENGTH TO EVEN MULTIPLE OF REFNUM/QQ
REFNUM → ECHNUM
ECHNUM REFNUM
SEG > 2MM+1
NO
MM TOO LARGE FOR SEGMENT LENGTH
TYPET
YES
FAULT
11

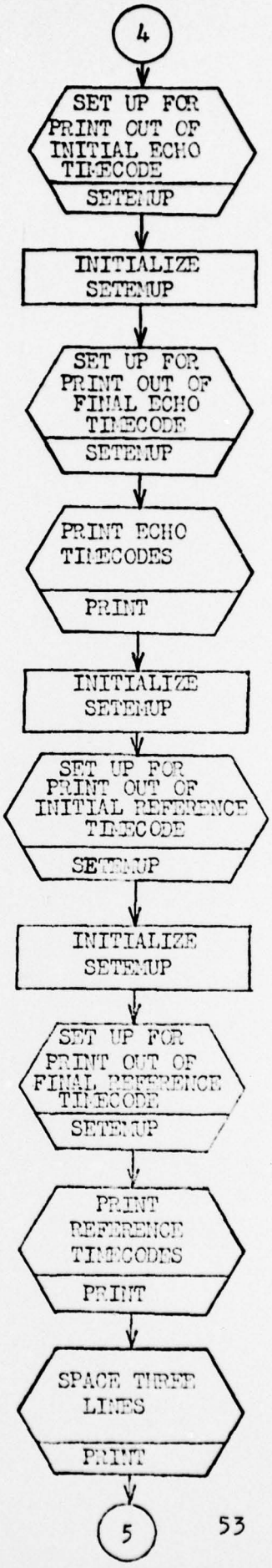

11
COMPUTE JJ AND CC
ALL DATA IN CORE
NO
SEGMENT ALL IN CORE
YES
14
YES
NO
INITIALIZE GENCORR
CORRELATION IMPOSSIBLE SEGMENT TOO LONG
TYPET
COMPUTE ONE SEGMENT OF CROSS-CORRELATION
GENCORR
SPECIAL EXIT
ALL SEGMENTS COMPUTED
NO
YES
8

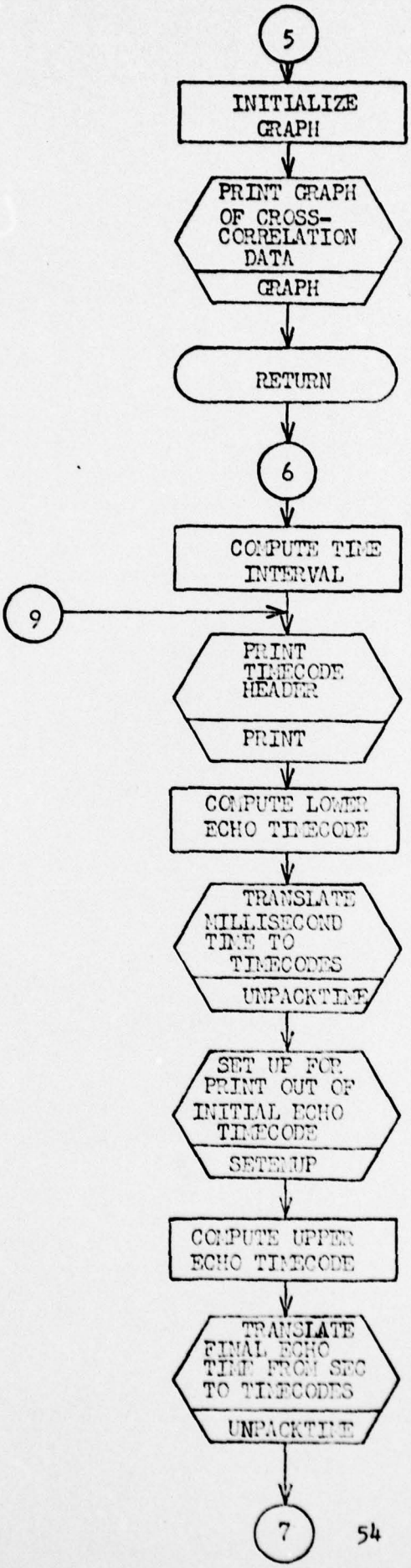

14
INITIALIZE
GENCORR
IS UPDATE
NECESSARY
NO
YES
READ IN MORE
ECHO SAMPLES
AS AVAILABLE
UPDATE
COMPUTE
ONE SEGMENT
OF CROSS-
CORRELATION
GENCORR
ALL
SEGMENTS
COMPUTED
NO
YES
8

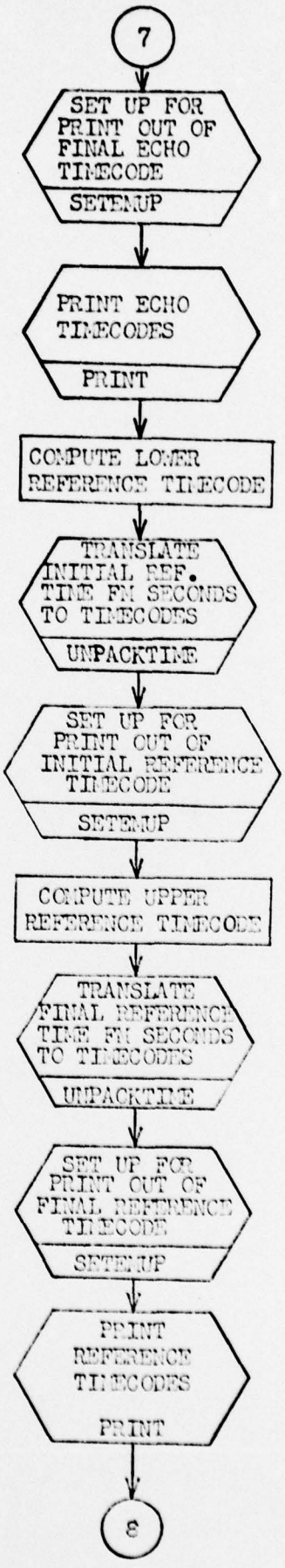

ENTER
POSITION OUTPUT TAPE TO THE BEGINNING OF THE DATA
TAPE
READ ONE RECORD OF OUTPUT DATA
TAPE
TOP OF FORM
PRINT
PRINT CROSS-CORRELATION HEADER
PRINT
PRINT DATE
PRTDATE
PRINT INDEX HEADER OF FIRST OUTPUT
PRINT
CLEAR B1, B2, B3
COMPUTE NUMBER OF LINES
1
PRINT ONE LINE OF DATA
2

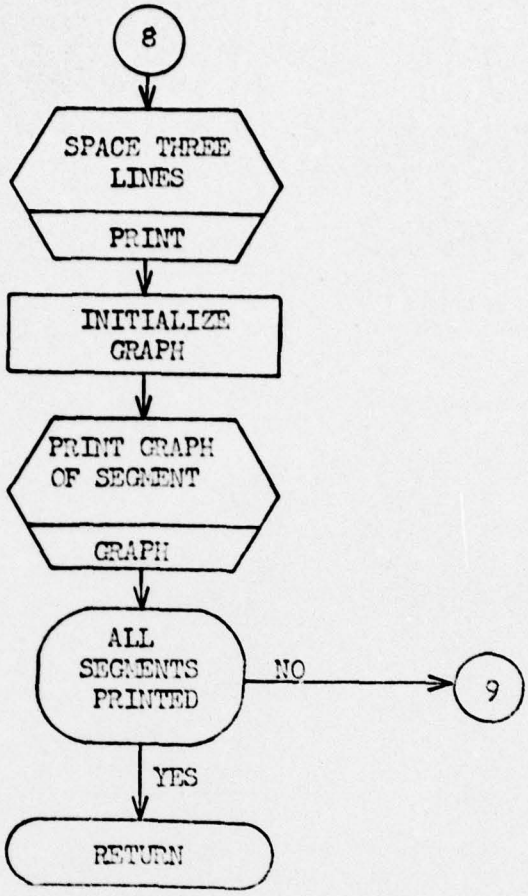

2
ALL DATA IN ONE RECORD
NO
YES
READ IN ANOTHER RECORD
TAPE
END OF FILE
1
NO
ALL LINES PRINTED
YES
KEY 1 SET
SET
RETURN
NO
POSITION OUTPUT TAPE TO THE BEGINNING OF THE DATA
TAPE
READ A RECORD OF DATA
TAPE
TOP OF FORM
TAPE
PRINT CROSS-CORRELATION HEADER
PRINT
3


3
PRINT DATE
PRTDATE
PRINT SIGNAL AND REFERENCE FILE NUMBERS
PRINT
PRINT SIGNAL AND REFERENCE TAPE UNIT NUMBERS
PRINT
PRINT QQ AND MM VALUES
PRINT
CONTINUOUS MODE
NO
PRINT SEGMENTED OPTION HEADER LINES
PRINT
YES
PRINT CONTINUOUS OPTION HEADER LINES
PRINT
PRINT HEADER LINES COMMON TO BOTH OPTIONS
PRINT
CONTINUOUS MODE
NO
6
YES
PRINT TIMECODE HEADER
PRINT
INITIALIZE SETEMUP
4


4
SET UP FOR PRINT OUT OF INITIAL ECHO TIMECODE
SETEMUP
INITIALIZE SETEMUP
SET UP FOR PRINT OUT OF FINAL ECHO TIMECODE
SETEMUP
PRINT ECHO TIMECODES
PRINT
INITIALIZE SETEMUP
SET UP FOR PRINT OUT OF INITIAL REFERENCE TIMECODE
SETEMUP
INITIALIZE SETEMUP
SET UP FOR PRINT OUT OF FINAL REFERENCE TIMECODE
SETEMUP
PRINT REFERENCE TIMECODES
PRINT
SPACE THREE LINES
PRINT
5


5
INITIALIZE GRAPH
PRINT GRAPH OF CROSS-CORRELATION DATA
GRAPH
RETURN
6
COMPUTE TIME INTERVAL
9
PRINT TIMECODE HEADER
PRINT
COMPUTE LOWER ECHO TIMECODE
TRANSLATE MILLISECOND TIME TO TIMECODES
UNPACKTIME
SET UP FOR PRINT OUT OF INITIAL ECHO TIMECODE
SETEMUP
COMPUTE UPPER ECHO TIMECODE
TRANSLATE FINAL ECHO TIME FROM SEC TO TIMECODES
UNPACKTIME
7


7
SET UP FOR PRINT OUT OF FINAL ECHO TIMECODE
SETEMUP
PRINT ECHO TIMECODES
PRINT
COMPUTE LOWER REFERENCE TIMECODE
TRANSLATE INITIAL REF. TIME FM SECONDS TO TIMECODES
UNPACKTIME
SET UP FOR PRINT OUT OF INITIAL REFERENCE TIMECODE
SETEMUP
COMPUTE UPPER REFERENCE TIMECODE
TRANSLATE FINAL REFERENCE TIME FM SECONDS TO TIMECODES
UNPACKTIME
SET UP FOR PRINT OUT OF FINAL REFERENCE TIMECODE
SETEMUP
PRINT REFERENCE TIMECODES
PRINT
8


8
SPACE THREE LINES
PRINT
INITIALIZE GRAPH
PRINT GRAPH OF SEGMENT
GRAPH
ALL SEGMENTS PRINTED
NO
9
YES
RETURN